Molecular Graphics: Protein Visualization

by

Belhadj-Mostefa, Khaled

a thesis submitted to the
Faculty of Science,
University of Glasgow
for the degree of

DOCTOR OF PHILOSOPHY

Department of Computing Science
University of Glasgow
May, 1990

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 – 1346

Declaration

The material presented in this thesis, except where stated below, is the product of my own original work carried out at the Department of Computing Science, University of Glasgow under the supervision of Dr Ron Poet and Dr James Milner-White. Parts of section $3 \cdot 5 \cdot 2$ and Appendix 3 represent work carried out collaborately with Dr Poet. Where the work of others is used, explicit reference is made in the text. No part of this thesis has been, or is being submitted for a degree at any other university.

To my Parents,

I owe them everything in this world

Acknowledgements

I am deeply indebted to my supervisors for proposing this study, for many valuable suggestions, and for much encouragement during its execution. Dr Ron Poet showed patience and without whose invaluable guidance and all-out support, this research could not have been concluded. Dr James Milner-White has also shown a great deal of support for me and a commitment to my work far exceeding all reasonable expectation. Every meeting with him has resulted in some growth in my research work and my working life.

I am grateful to Professor Dennis Gilles who gave me the opportunity to study for this degree. I would also like to acknowledge Professor Malcolm Atkinson for his particular interest in this work and for the confidence boosting assistance he gave me when I needed it the most. Dr Rob Irving is also well acknowledged for his time devotion, kindness and warmth whenever I needed to see him. My brother and sister, Zoubir and Soraya, who showed a deep level of concern and support for my research are acknowledged with similar feelings for theirs. I also like to thank my colleagues Abdellatif, Djamel, Roslan, Sembok, Saeed, Riaz, Gebre and others for many happy and unforgettable moments that we all shared together as students in this department.

Above all, I would like to express my hearty thanks to my wife, Lamia, who endured my having to be away from her for so long a period.

Financial support and study leave were provided by the Algerian Ministry of Higher Education which is appreciated and gratefully acknowledged.

Table of Contents

List of Figures

Abstract

The work in this thesis describes the development of new ways of picturing proteins that assist in understanding both their three-dimensional structure and the relationships between it and their sequence.

Proteins are extremely complex three-dimensional objects that need to be examined at several levels of structure: overall shape, chain structure and folding, secondary structure, atomic details and sequence. Many further levels can be added. It is difficult to display all such information in one picture for these very large molecules, because of the huge amount of detail involved. In the present work, I have developed novel forms of display that allow the user to focus on features appropriate at the different levels. The window management techniques incorporated in the Colour Whitechapel Workstation (CG-1) are employed so that the multi-level displays can be manipulated interactively and displayed simultaneously.

A special feature of my system is the way I have developed new strategies for indicating depth. A combination of well-known techniques for depth perception with some new ones developed in this work, have made it possible to create models of proteins which stimulate the imagination of biochemists, while minimizing time delays for display or manipulation of the images. These techniques can be applied generally over various fields of Computer Graphics such as CAD/CAM systems.

This new type of computer generated picture allows the three-dimensional hydrogen bond patterns of loops and secondary structural features to be displayed. A recursive algorithm has been developed that differentiates between various sorts of hydrogen bond patterns.

A further novel type of picture developed in the present work displays hydrogen bonds in proteins in relation to the primary structure, rather than, as in more conventional pictures, in the context of the three-dimensional structure. I refer to these displays as diagrammatic views. For such views, two different algorithms are described (one for the inter-mainchain hydrogen bonds and the other for the sidechain — mainchain ones). In the pictures, hydrogen bonds are drawn diagrammatically so that all can be clearly seen and all secondary structure features and loop motifs portrayed. No other known computer-generated pictures provide such comprehensive information about this aspect of proteins.

The research provides a new tool to help prediction of three–dimensional structure from sequence. It is of importance because it allows the relationship between three-dimensional structure and sequence to be readily investigated.

Chapter 1

INTRODUCTION

1·1 PROTEIN STRUCTURE DETERMINATION

1·1·1 The Problem

Many problems of modern biology are concerned with the relationship between biological function and molecular structure. It is only when the three-dimensional structures of all molecular components of a biological system are understood and appreciated, that biochemists are in a position to answer the fundamental questions.

The critical determinant of the biological function of a protein is its conformation. This is defined as the three-dimensional arrangement of the atoms of a molecule. It can be determined by means of a technique known as X-ray crystallography [KENDR 61]. Despite the fact that the

structure of more than two hundred and fifty proteins have been derived from it, the method is tedious and time consuming. On average, 5 to 10 man-years are required for one. On the other hand, the work of Anfinsen [ANFIN 73] demonstrated for the first time that the three-dimensional structure of a protein is specified by its amino acid sequence. This experiment was based on the fact that after being unfolded by treatment with chemicals and these chemicals then removed, a protein appears to regain its natural form and thus its functional activity. This experiment showed that the information needed to specify the complex three-dimensional structure of a protein is contained in its unique amino acid sequence [STRYE 75]. Chemical techniques are capable of revealing this sequence. The technique of DNA sequencing, which is relatively easy to perform with the advent of genetic engineering has led to a rapid growth in the number of known protein sequences, but not their structures.

1·1·2 The Technique

From the above, it might be expected that it should be possible to predict theoretically the three-dimensional structure of a protein from its sequence. This is an active area of research in molecular biology (for review see [STERNB 83] & [TAYLO 87]).

Currently, opportunities in the structure prediction field have been transformed by four main factors:

1- The realization that proteins exist in families, and that the three-dimensional structure is much more highly conserved than the sequence, together with the availability of sequence data for

several members of such families.

2- The availability of sequence data for several proteins belonging to the same family.

3- Greater knowledge about protein three-dimensional structure.

4- Advances in computer technology.

Each of these alters the nature of approaches that can be used for prediction.

1·1·3 The Tool

Computer graphical methods of visualizing structures are important tools in this research as well as complementing laboratory experiments. They are experimental tools in their own right and play a major role in molecular biology as well as in education. With some practice of an appropriate molecular graphics system, a user can view molecular models interactively on a screen. They can be manipulated as the user chooses. Because the object under consideration has no analogue in the user's experience of real life, various representations of proteins have been derived.

Our work consists of developing new ways of displaying proteins in order to get insights about the different levels of their structure: overall shape, chain structure and folding, sequence, secondary structure and atomic details. Many further levels can be added. It is difficult to display

all such information in one picture for these very large molecules because of the huge amount of detail involved. I have concentrated on developing relatively simple and informative pictures that focus on the relevant features that are appropriate at the different levels. The pictures can be manipulated interactively and displayed simultaneously using windowing technology.

1·2 TERMINOLOGY

This section defines some general terms used in this and subsequent chapters. Proteins are very large molecules composed of thousands of atoms. They contain carbon (C), hydrogen (H), oxygen (O) and nitrogen (H) atoms; some also contain phosphorus (P) and/or sulphur (S). They are build up of *amino acid units*. There are twenty major naturally occurring ones and some minor ones. The number of amino acid units in a protein may range from a few to thousands, but they are mostly large molecules with, typically, 50 to 2000 amino acids; some contain only a selection of the possible amino acids, but most contain all of them. The particular amino acids present and the order in which they occur determines the individuality of the protein. Amino acids have a common basic structure consisting of an amino group (NH_2), a carboxylate group (COOH), a hydrogen atom (H), and a distinctive R group bonded to a carbon atom called the alpha-carbon or C_α. The R group is referred to as the *side chain*. The basic formula is shown in Figure 1 · 1. Some side chains are uncharged, others may be either positively or negatively charged.

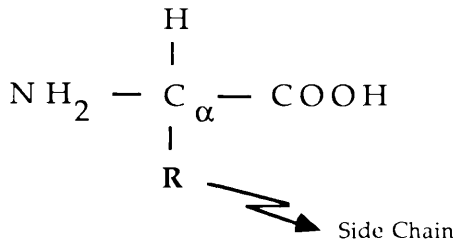


FIGURE 1 · 1 Common Structure of Most Amino Acids

The simplest amino acid is glycine where R is a hydrogen atom; in alanine R = CH_3 and in valine R = C_3H_7. In more complex amino acids, R contains additional methyl, hydroxyl, thiol or carboxyl groups or a ring structure. One amino acid that does not conform exactly to this common structure is "proline", shown in Figure 1 · 2, where the side chain is bonded to the amino group as well as to the alpha-carbon, thereby forming a cyclic structure.

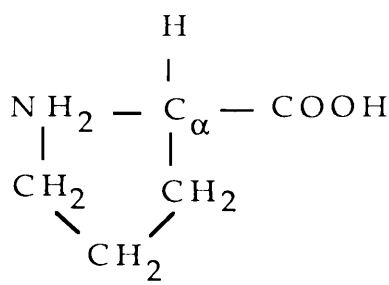


FIGURE 1 · 2 Proline

When two amino acids are joined together in a protein molecule, a covalent bond is formed between main chain atoms as indicated below. The new bond is called a *peptide bond* (see Figure 1 · 3). This process can be repeated to form a long chain : a *protein molecule.*

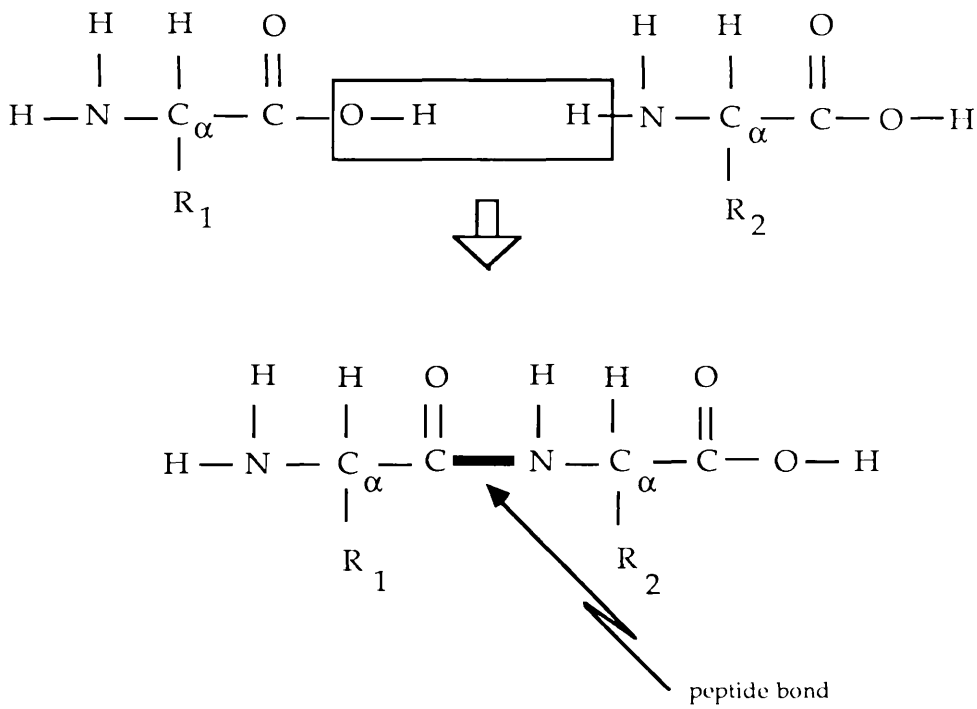


FIGURE 1 · 3 Formation of a Peptide Bond

Although all its bond lengths and angles are fixed, a protein molecule is not rigid because some of its bonds can rotate. The amount of rotations about each bond determines the three-dimensional structure of a protein. The bonds $C_\alpha - N$ and $C_\alpha - C$ are single bonds. Rotation about these bonds are designated by the angles Ψ and Φ, which are called the *dihedral angles* [RICHA 81]. The only exception is the peptide bond which does not rotate because this link has partial double-bond character.

Moreover, the equivalent angle between the N and C atom is nearly equal to 180 degrees. Those constraints causes the six-atom group associated with every peptide bond to form a rigid, planar structure called a *peptide unit*. It is depicted in Figure 1 · 4.

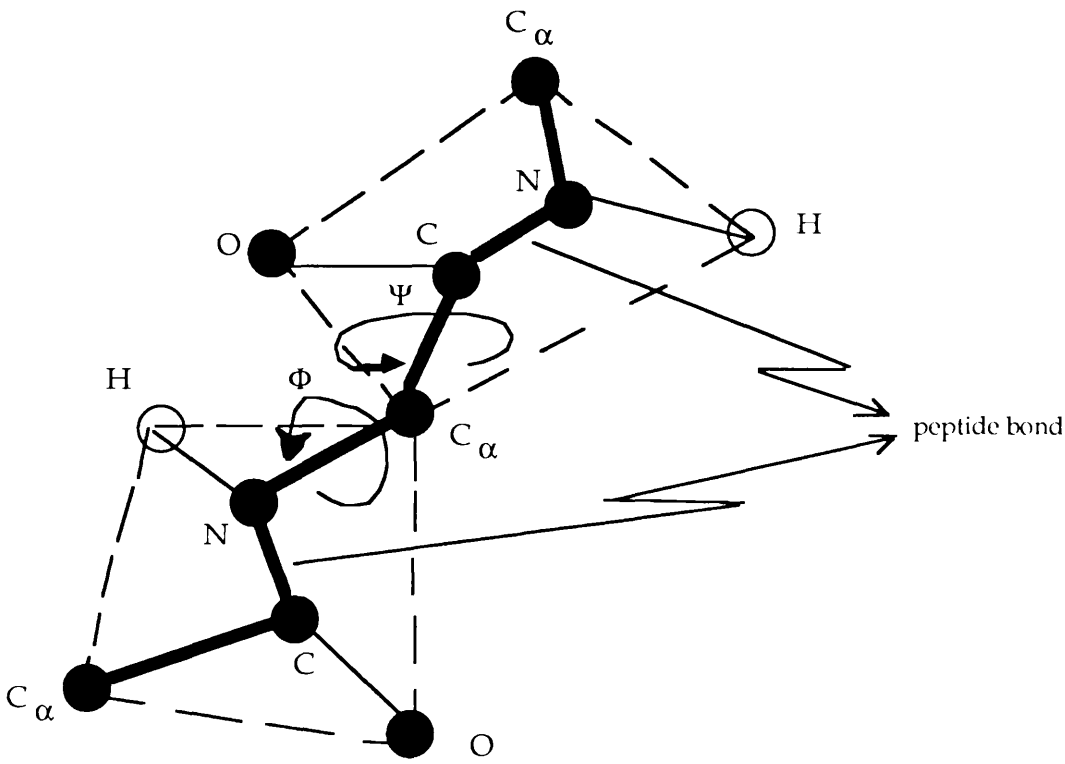


FIGURE 1 · 4 Two Peptide Units

An amino acid in a protein joined to other modified acids may be called a *residue*. Figure 1 . 5 shows how residues are joined by peptide bonds to illustrate the formation of a protein. These residues form a *polypeptide chain*. In some proteins, the polypeptide chain may be cross-linked at some places by disulphide (—S—S) bonds found in pairs of "cysteine" side chains. The *main chain* (or sometimes called the

backbone) of a protein is the substructure containing the N, H, C_α , O and C atoms of the common part of its residues. The nitrogen atom at one end of this main chain, the *N-terminus*, retains both hydrogen atoms found in the acid from which the residue was derived. Likewise, the carbon atom at the other end of the protein, the *C-terminus*, retains its hydroxyl group.

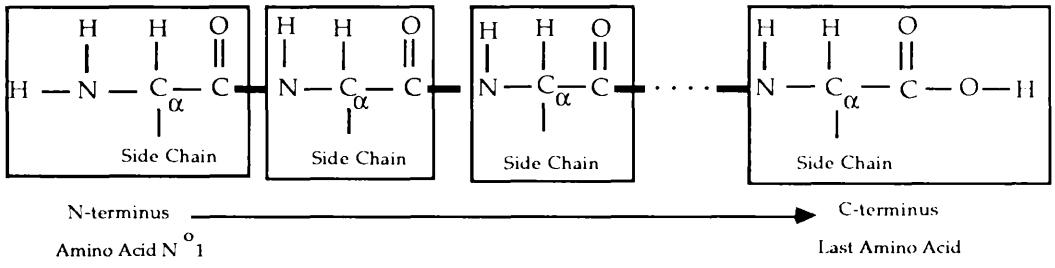


FIGURE 1 · 5 : A Polypeptide Chain

Some sections of these amino acid chains happen to fold up to form characteristic helical shapes called *alpha helices*. An alpha helix is stabilized by *hydrogen bonds* between the NH and CO groups of the main chain. The CO group of each amino acid is hydrogen bonded to the NH group of the amino acid that is situated four residues ahead in the linear sequence. There are other such features (*parallel* and *antiparallel beta sheets, beta turns etc...*) which will be discussed more thoroughly in section 3 · 6.

Many protein molecules are composed of a single polypeptide chain. They are called monomers. Others may be composed of 2 (dimers), 4 (tetramers) or more chains, which may be identical or different; the chains are held together either by noncovalent forces or linked together

by disulphide bonds. Each polypeptide chain in such proteins is called a *subunit* which is folded to form a globular structure. Haemoglobin, for example has 4 subunits, 2 called alpha and 2 beta [PERUT 64]. A *domain* is a folding unit within a subunit.

Several levels of structure need to be examined in discussing the architecture of proteins. First, the *primary structure* which is the sequence of amino acids. Next, the *secondary structure* refers to regions of polypeptide chains that are involved in alpha helices, parallel and antiparallel beta sheets. *Supersecondary structure* refers to groups of secondary structure units. *Tertiary structure* refers to the overall folding of groups of amino acid residues. Proteins that contain more than one subunit display an additional level of structural organization, namely the *quaternary structure*, which refers to the way in which the subunits are packed together [STRYE 75].

1·3 MEDICAL IMPORTANCE

1·3·1 Design of New Drugs

The detailed understanding of protein structure that is bought about by molecular graphics and structure prediction has several medical benefits. One is that it is helpful in designing new drugs. Most drugs interact with proteins by binding to them; so the best way, currently, to design new drugs is to examine the shape of the site on the particular protein at which it acts and to synthesize a new drug molecule that fits in the site.

1·3·2 Vaccines

A second area concerns vaccines. Vaccination essentially involves introducing a foreign protein into the body so that the immune system is primed to recognize that protein as part of an invading organism. Instead of using the whole protein it is now possible to inject a small length of the polypeptide chain. The problem however is to determine which part of a protein is immunogenic. The best way to do this is to examine the protein by eye and see which parts lie at the surface.

1·3·3 Protein Engineering

A third category of medical benefit lies in the field of protein engineering. Using genetic engineering techniques, it is now possible to create proteins with altered sequences and therefore new properties. It is obvious that an understanding of what to alter is only possible with a knowledge of the three–dimensional structure of the protein and some ability to predict the effects of sequence on structure. This new technology will probably have a wide ranging effect in medical science and biotechnology. One area in which more immediate effects may be anticipated is in creating new antibody molecules with properties that make them suitable for injection directly to humans. Some progress has already been made in this area and it may for example be possible to create antibodies that are a mixture of human and animals, the main part deriving from humans and the specific part being derived from an animal. Thus the specific part of the antibody can be elicited in animals which is useful because human is ineffective for this purpose while the main part of the antibody, being mainly of human origin, does not give rise to the adverse effects normally found on injection of animal

proteins.

1·4 SCOPE OF THE THESIS

This research is intended to develop new ways of picturing proteins with the aim of assisting the understanding of their three–dimensional structure and of the relationship between sequence and three-dimensional structure.

An important chemical feature governing the shape of proteins is the presence of many inter-mainchain hydrogen bonds. There are two main arrangements of such bonds that correspond to the two major types of secondary structure found: one is the alpha helix where the chain forms a helix, the other is the beta-sheet where extended regions of chain run parallel or antiparallel to one another (see section 3 · 6). Several representations used to display these secondary structures are described thoroughly in Chapter 2, together with an analysis showing a deficiency in currently available systems.

This work describes a way of displaying polypeptide chain structures where the research reported by Milner-White and Poet has been the starting point (see [MILNE 85], [MILNE 87a], [MILNE 87b] & [POET 86]). Pictures of the polypeptide chain with either the inter-mainchain hydrogen bonds, or those involving sidechain-mainchain bonds can be displayed and manipulated interactively. Colour coding has been used to differentiate between the different types of hydrogen bond found.

A recursive algorithm for the inter-mainchain hydrogen bond pictures has been developed (see section 5 · 4 · 1 · 2) that differentiates

between various sorts of hydrogen bond patterns. This algorithm is particularly useful for detecting the number of parallel beta-sheets in a protein (used subsequently to generate the diagrammatical pictures described in the next paragraph). It also distinguishes between hydrogen bonds belonging to a parallel beta sheet or an anti-parallel one, which was not easily differentiated in the early pictures of the project. Consequently, the colour coding of hydrogen bonds in these pictures was more revealing. Atomic details of a particular region are also available for display and manipulation.

The question of how to represent depth is of special concern in the present study and ways of visualizing three-dimensional objects on a two-dimensional screen without disturbing the concentration of the viewer has been one of our goals. I have avoided the use of expensive and time consuming software tools such as stereoscopic views or rocking mechanisms (rotating back and forth the object through an axis where the three-dimensional effect disappears as soon as the motion ends) to enhance the three-dimensional perception.

Certainly speed of manipulation of the model is also a factor to consider, but is less important since manipulation of the model is seldom required in the present application. Concentration on stationary pictures is mostly needed. However, smooth motion of the object under consideration, in terms of speed, would also avoid disturbance in the viewer's concentration.

I developed some new techniques described briefly in the next section, which help enhance the three-dimensional illusion for the user. This combination of techniques has made it possible to create a model image of a protein which stimulates the imagination of biochemists, while avoiding on one hand expensive software tools and on the other

hand large time delays for the display or manipulation of the images.

A further new computer-generated graphical display technique for proteins is also described (see section 4 · 7), in which the backbone is extended as a straight horizontal line and where the inter-mainchain hydrogen bond structure (or the sidechain — mainchain one) is related to the sequence. One advantage of this type of display is that every hydrogen bond is clearly seen since it cannot be obscured by atoms in front of it, as happens occasionally in the displays described above. Another is that the relationship between bond pattern (and consequently the three-dimensional structure) and the sequence is evident. Two slightly different algorithms are described, one for the inter-mainchain hydrogen bonds (section 5 · 4 · 5 · 1) and the other for the sidechain — mainchain ones (section 5 · 4 · 5 · 2), that allow the hydrogen bonds to be drawn so that they can all be clearly seen and characteristic features portrayed. These displays are especially useful for representing inter-mainchain hydrogen bonds exhibiting characteristic patterns that we collectively call loop motifs.

Several examples are illustrated, including the protein wheat germ agglutinin. This protein is composed of 2 subunits and each subunit consists of 4 highly homologous domains that are relatively lacking in secondary structure and yet have a large number of loop motifs. One of the conclusions of this work is that we are able to propose that hydrogen-bonded loop motifs are conserved in much the same way that secondary structure is, and that it would be better to include loop motifs in the secondary structure category. Another conclusion of this work is that it provides a new tool to help prediction of three-dimensional structure from sequence because it automatically relates structure to sequence.

The work described above has been brought together to give birth to

a new integrated molecular graphics software package called PHD (Protein Hydrogen-bond Display), currently implemented on the Whitechapel Colour Workstation CG1 and which can be easily migrated to a different hardware. The interface is mostly menu-driven. The system provides a working environment for the biochemist allowing him to manipulate models of known protein three–dimensional structure. Interesting sections of proteins can be selected, showing chain structure, atomic details or diagrammatic views. The latter are useful for making comparisons of sequence versus structure conservation because two such diagrams can be displayed in the same screen picture. At the end of a session, the work may be saved.

The strategy in picturing proteins is based on the idea of multiple simplified displays where only important information is displayed so that the investigator is not swamped by unnecessary detail. These pictures can be manipulated interactively and displayed simultaneously, taking advantage of window management techniques incorporated in the CG1.

1·5 THREE-DIMENSIONAL DEPTH EFFECT

The visualization of these three-dimensional objects gives rise to the question of how to indicate depth in our pictures. One prerequisite is to be able to see through the object displayed. This is achieved by using line segments to represent the polypeptide chain pictures as well as the atomic detail ones. Various three-dimensional techniques have been applied in these pictures in order to get a three-dimensional effect:

- depth sorting : line segments are drawn in depth order with the farthest being drawn first.

- intensity depth cueing : line segments near the viewer appear brighter than those far away.

- brush shapes : Various drawing brush shapes have been implemented to indicate the relative depth of two consecutive line segments. It gives the appearance of a line segment being in front or behind its neighbouring one (see section 4 · 5 · 7 · 4 for further details).

- tramlines : A much faster technique for drawing *"tramlines"* than the one of polygon filling used by Poet & Milner-White [POET 86] is presented. The relative depth of two line segments is indicated by a bridge effect when they cross. This is especially effective if the line segments are very close to each other in depth; the depth cueing effect alone is insufficient since the line segments will have the same colour intensity (see section 4 · 5 · 7 · 3 and section 5 · 4 · 2 for further details).

1 · 6 OVERVIEW OF THE THESIS

Chapter 2 concerns the literature review and discusses various aspects of some systems used all around the world to model proteins. A closer look is made on the way these systems handle protein representation, the techniques they use to manipulate objects and the communication between the system and the user. A final section will be devoted on criticism and the difficulties encountered by these

systems.

Input to the PHD software package are data extracted from the Brookhaven PDB. These data, to be used in the PHD protein visualization package, need to be preprocessed. Several programs are described in Chapter 3 which preprocesses the relevant data and store them in various files. The remainder of the chapter is concerned with some definitions and criteria for assigning secondary structures from the PDB (see Section 3 · 6). The definitions deal with hydrogen bonds in alpha-helices, beta strands and various turns. They are needed latter, in Chapter 5, when an algorithm is presented which distinguishes between various types of secondary structure features.

In designing the PHD protein visualization package, the first stage is to consider the user interface design. The factors to be considered in the design of the user interface — specific operations made available to the user, output organization and how the package is to be documented, presented, and manipulated by the user — are thoroughly explored and described in Chapter 4.

Details of internal structures, data structures design and manipulation, algorithm descriptions needed for the PHD package, are all presented in Chapter 5.

The PHD system, described in the previous three chapters, is tested in Chapter 6 by examining domain homology for a particular protein and by trying to get some insights about the mystery of protein evolution. Results of the analysis suggest an alteration in the definition of secondary structures.

Conclusions and future research are discussed in Chapter 7. It

discusses the overall contribution of the present work; this includes the status of PHD as a means of providing an integrated system for displaying known three-dimensional protein structures as well as sequences related to the structure.

Chapter 2

SURVEY OF RELATED WORK

2·1 INTRODUCTION

After a short overview in section 2·2·1 of early methods used to model molecules and a mention of some of the difficulties encountered in them, a description of the first effective application of interactive computer graphics to molecular structure is given in section 2·2·2. That developed at UCSF/Computer Graphics Laboratory was a prototype for many of the molecular graphics laboratories now operating, and is described in section 2·2·3. Some further work pertinent to protein display is described in the sections after that. Several features closely related to my study emerge from these systems (i.e. representation of protein models, interaction with the user). Each system will be described in terms of its hardware configuration, language implementation, software capabilities and graphics techniques adopted, user interface and eventually its line of extension in the future. The variety of these

systems, chosen from many part of the world, shows the interest in the scientific community toward molecular graphics. An analysis showing the deficiency in currently available systems is made in section 2 · 3.

2·2 MOLECULAR GRAPHICS HISTORY

2·2·1 Early Devices

A molecular biologist's understanding of a molecular structure is usually reflected in the ability to construct a three-dimensional model of it. While it is difficult to represent those structures on paper, especially for large molecules such as proteins, the construction of physical models through balls and sticks is even more tedious and time-consuming and storage of such structures is awkward.

Another major problem is that the model and the actual list of coordinates are not necessarily closely related, especially after manipulation of the model. Lastly, it can be frustrating if the model sags. So a need for other techniques had to be devised. Computer graphics was the key and the indispensable tool which provided solutions to these problems.

With it, the display and the data are directly related. Storage of prior configurations is simple, and pieces do not fall off.

2·2·2 Project M A C

The history of Molecular Graphics started with project MAC[1], led by Cyrus Levinthal [LEVIN 66]. It was the first appearance of an application of interactive computer graphics to molecular structure representation. A calligraphic display system was designated to permit direct interaction of the investigators and a molecular model that was being constructed by the computer.

In spite of the limitations of this early system, real-time three-dimensional rotation of the displayed object was possible for the first time. The drawings of the molecules have a line segment for each bond and could be modified in real-time. At that time, where the number of vectors that could be displayed on the screen was rather limited, the interaction controlled by manipulating various input devices (i.e. keyboard, light pen and knobs) is still of interest, nowadays.

One of the fascinating aspects of this fruitful project was, as Morffew pointed out [MORFF 83a], that it was aimed at finding a way of folding proteins in their final three-dimensional structure, but did not succeed mainly because of the lack of scientific breakthroughs at that time. As an example, the approach taken to predict the structure of a protein in advance of its determination of X-ray analysis, was based on the supposition that the folding starts independently in several regions of the molecule and that the first structural development was the formation of a number of segments of alpha helix which in turn interact with one another to form the final molecular structure.

---

[1] Standing for Multiple Access Computer and was the name of the mainframe time–sharing project at M.I.T. Boston, Massachussets.

This project boosted molecular graphics. In these last two decades, great advances have been made in the capabilities of graphics systems. Furthermore, the rapidly decreasing cost of hardware made it possible for many laboratories to afford sophisticated systems.

2 · 2 · 3 U C S F / Computer Graphics Laboratory

Robert Langridge, one of the researchers at MAC project, helped the development of the Computer Graphics Laboratory at Princeton University[2]. The basic configuration was the first calligraphic Evans & Sutherland system, the LDS-1, which provided in high-speed, digital hardware 4 x 4 matrix multiplication and concatenation, three-dimensional clipping, and perspective [LANGR 74]. It was interfaced to a Digital Equipment Corporation (DEC) PDP-10. This hardware innovation was of great importance to display a three-dimensional object on a two-dimensional screen as a series of projections are calculated as the object is rotated. The coordinate transformations require several multiplications and additions for every end-points and the key to smooth representation of complex objects is to implement them in hardware. This is most efficiently done with homogeneous coordinates, using a 4 x 4 matrix representation of the transformations such as rotation, translation, scaling and perspective [NEWMA 79]. Those matrices may then be concatenated and handled as a single entity.

An interactive display program Computer Aided Analysis of Protein Structure (CAAPS) was designed and provided many of the tools needed for protein engineering, such as amino acid replacement, bond

---

[2] The Princeton University Computer Graphics Laboratory was established by a grant to Langridge in 1970 from the division of Research Ressources of the National Institute of Health.

manipulation, monitoring van-der-Waals contacts[3] between parts of the protein structure undergoing modification, and stereoscopic viewing [LANGR 74]. The aim of this system was to enable a user to study any part of a large number of protein structures in any details.

Later, Langridge went on at San Francisco and created the UCSF/Computer Graphics Laboratory (CGL). The display available was a black-and-white Evans & Sutherland Picture 2 calligraphic display (PS2). Besides all the previous hardware transformations, it was capable of performing not only real-time clipping on top, bottom and sides of the picture space but also in the depth axis. This is associated with a "viewing pyramid" which includes a hither (near) boundary, yon (far) boundary, and eye position [NEWMA 79]. The intensity of the cathode-ray tube (CRT) electron beam may be controlled in proportion to the distance between the hither and yon planes, giving a strong sense of depth to the picture. The host processor, a DEC PDP 11/70 computer was running under the UNIX[4] time-sharing operating system to support the high performance graphics display system for displaying three-dimensional molecular models [FERRI 80]. A large variety of high programming language is available for use and switching from one language to the other is easily performed depending on a particular application. However some obstacles were encountered primarily because the Evans & Sutherland provided no support for their graphics hardware under UNIX. A decision has been made to rewrite all the graphic subroutine package that Evans & Sutherland did provide, from assembly to C language and this made the day of the first UNIX based

---

[3] Atoms have a particular radius associated with them which is called the van-der-Waal radius. Each atom is represented as a sphere, and the spheres for 2 non-bonded atoms are not allowed to come into contact with each other.

[4] UNIX is a Trademark of Bell Laboratories

three-dimensional graphics package for the PS 2.

A major molecular graphics system has been developed in this environment : Molecular Interactive Display System (MIDS). It was able to accommodate the new developments in colour displays [LANGR 81]. It introduced new approaches to molecular graphics, particularly the display of that part of the molecule surface which is accessible to other molecules that is the "solvent-accessible surface" defined by Lee and Richards [LEE 71] and improved on by Richards [RICHA 77]. The idea is the following: if you take a protein and place a water molecule as close to that protein as possible, then rolling the water molecule all over the protein, you would get this solvent-accessible surface. This surface is then used not only to see how close a water molecule can approach, but also to get an idea on how close other molecules could approach.

The next product, the Molecular Interactive Display and Simulation (MIDAS) emphasized interactive display and manipulation of proteins and nucleic acids [FERRI 84]. The introduction of a PS2 colour monitor had the advantages of providing bright, high-resolution, and full-colour pictures. Hardware capabilities such as depth cueing, perspective, clipping and real-time stereo are incorporated. Dot surface representations of the Richards' surfaces were introduced ([LANGR 81] & [LANGR 84]) and an algorithm to calculate the molecular surface was developed by Michael Connolly ([CONNO 81], [CONNO 83a] & [CONNO 83b]). The basic approach in the method is to draw a surface not as a solid surface, but as a series of transparent dots on the surface. Internal cavities corresponding to the inner molecular surface of the protein are properly represented by the use of hither and yon clipping planes to clip away the dots at the front and the back. Close-up view of a binding site can be obtained by real-time rotation, scaling and clipping. The use of dots covering a surface in combination with the depth cueing and colour gives excellent three-

dimensional representation of molecular surfaces using a line-drawing display. A procedure was latter developed by Bash [BASH 83], dealing with interior atom removal. This technique had the significant advantage that the space-filling representation is retained during interactive real-time bond rotation. All of these methods have proved to be extremely useful in enzymology, immunology, virology, molecular pathology and the study of protein-ligand and protein-protein interactions [CURREN 86]

Since a huge amount of data had to be managed, a hierarchical data base management system was designed at the core of MIDAS by Ferrin and al. ([FERRI 88a] & [FERRI 88b]). Thus, this system provides a flexible tool for the study of small and large molecules and their interactions, taking full advantage of available interactive three-dimensional colour display capabilities.

2·2·4 XRAY: Interactive Molecular Display and Modelling System

Graphics hardware employed by Feldman and his co-workers [FELDM 73] at the National Institutes of Health in Bethesda (Maryland – U.S.A. –), to develop their interactive molecular display and modelling system (XRAY), consists of two vector graphics display systems, an ADAGE AGT-30 and a DEC 340 graphics terminals, driven by a DEC PDP 10 time-sharing minicomputer. The software is written in PDP 10 assembly language which does not make it readily portable.

In these early stages of molecular graphics, where large machine dependent systems are used to interactively display and model proteins (see [LEVIN 66] & [BARRY 69]), XRAY is somewhat versatile as its hardware configuration permits a user to run any program of the system he wishes to display and manipulate, in any remote terminal of the two

types listed above.

Simple stick and ball and stick models can be displayed in a stereoscopic view on these monochrome screens. Rotation of the model about three orthogonal axes, as well as zooming on a particular region and labelling of amino acids are possible. Some modest model building facilities are also available that allow the user to modify particular residues by rotating about bonds or by replacing any side chain by another one. Plots showing a stereoscopic view of the picture are also generated.

User control of the system is by means of dials and few switches and are available only on the ADAGE devices.

The system has evolved very quickly especially with the rapid growth of computer technology. Ultimately, an Atlas of macromolecular models ([FELDM 76] & [FELDM 80]) has been made available to the public which is a powerful document in studies of proteins.

2·2·5 ALPHA : An Interactive Protein Model Building Program

The Molecular Modelling System ALPHA developed at the Ashahi Chemical Industry Co. Ltd in Shinuoka, Japan, by Toma [TOMA 87a] is based on an Evans & Sutherland Picture System 340 (PS 340) driven by a VAX 11/780 mainframe computer. It has both vector and raster displays. The vector display is used as a model building and a structure manipulation terminal while the raster display is used for filling-space models. The software is written in FORTRAN and the PS340 programming language.

The system is used not only for the study of proteins of undetermined structure (e.g. model building) but also to investigate proteins of known three-dimensional structure.

Representation of proteins is made only with the alpha-carbon model which has proved to be efficient for mainly three reasons. First, because it is preferable to deal with simple models than the detailed atomic ones which give a rather complicated image of the protein by making visual inspection difficult. Secondly, if interactive response is needed, the fewer data are given the faster and easier their processing is performed. Finally, flexibility in structure modification is made possible by assuming some realistic feature assumptions about the length between alpha carbons being constant.

On the vector display, real time rotation, scaling and translation are possible. Various orthographic views, depth-cueing and stereoscopy are available to give a more realistic three-dimensional image of the model. To modify the structure, first the user has to point to an alpha-carbon atom then one of the two part of the polypeptide chain can be rotated freely around the chosen point using dials. This is much the same idea as the one used in the MAC project [LEVIN 66] twenty years ago and described earlier in section 2 · 2 · 2.

An amino acid information colour-coded CPK-like model corresponding to an alpha-carbon model on the vector display can be shown on the raster display. Options on colour codes are available according to the chemical characteristics of the amino acids. This is relatively helpful for illustrating structural features of proteins. Cross-sectional views of the molecule can be displayed and are useful not only in the evaluation of the building structure but also in the analysis of known structures.

User control of the system is accomplished via a light pen to point at elements on the screen, dials to manipulate the model, and function buttons for the choice of colour codes.

The project had some further extensions and allows the investigation of protein-protein inter and intramolecular interactions [TOMA 87b]. It is basically a two or more protein structure problem were the alpha-carbon model was again used and proved to be a very effective model.

2 · 2 · 6 VENUS : Various Representations of Protein Structures

The hardware comprising the system VENUS[5] developed by Iga and Yasuoka [IGA 84] at Osaka University, Japan, consists of a N.E.C. N6960 raster graphics colour display which is connected by a RS232C communication line to a host computer, an ACOS 850 produced by Seiko Electronic Industries. The display, 1024x1024 in resolution, is equipped with a 760 kbytes memory buffer and four frames of digital video memories each having the same size as the number of pixels on the display. The programs are written in FORTRAN and uses the graphics package GDSP/SPLOT.

The system is used to help a researcher to get a better understanding on the folding of globular proteins as well as their secondary structures by displaying various representations of them.

---

5 VENUS was inspired by PLUTO, a system developed by Motherwell [MOTHE 78] which was used to prepare figures on a digital plotter. Its name has been chosen as the name of the planet in the solar system following PLUTO.

The majority of these representations involves the alpha-carbon backbone. Among these models, we have the wire model where successive alpha-carbon atoms are connected with lines, the conventional ball-and-stick model and several space-filling models representing the van-der-Waals surfaces.

Other representations less popular are also available for display. They include the pleat model where consecutive alpha-carbon atoms are connected by platelets. That is, a small line segment is created from each alpha-carbon atom, which is perpendicular to the plane passing through this atom and the two adjacent to it. The free ends of the line segments are connected sequentially to give a pleat model. If the coordinates of the alpha-carbon atoms are connected by a cubic spline, this will create the so called smooth-line model. Another variant would be the ribbon model where the smooth-line model is widened at turns. Manipulation of the model is performed by the so called *"pseudo-dynamic manipulation"*. For instance, in the case of a rotation around the x-axis, this is accomplished by preparing 36 models which are rotated in sequence by 10 degrees around the x-axis and stored in the memory buffer before being displayed either simultaneously or successively. Segments of the structure may be selected. Options are available for labelling atoms and residues, with or without sequence number. Although only seven colours are available, they seem to be adequately informative.

Interaction with the system is menu-driven. A menu is presented on the screen, and the user selects an option by synchronizing the movement of a pen on a tablet input device with a cursor displayed on the screen. Once the cursor reaches the desired option the pen is pressed. There are no button functions, joysticks, switches or knobs to manipulate the model under consideration. The pen is also used to select pieces of structure. All continuously variable user inputs controlling rotation,

translation, etc... come from the pseudo-dynamic manipulation.

The extensions of the project are toward molecular design and protein crystallography.

2·2·7 LESK'S MODEL : Schematic Diagrams of Protein Structures

The system developed by Lesk and Hardman ([LESK 82], [LESK 84], & [LESK 85]) at the Fairleigh Dickinson University, New Jersey (U.S.A.) is based on both a vector generator display and a raster display and are driven by an DEC VAX 11/780. The VAX is tightly coupled to an IBM 370/165 mainframe computer. The programs have been written in FORTRAN.

Although computer-generated pictures simulating wire models or space-filling molecular models may contain detailed information about spatial relationships of the atoms, the need for abstract representation has also proved to be informative for the investigation of tertiary structure and of homology between different proteins (see [RICHA 81]). Lesk and his team suggested a schematic-diagram approach to visualize proteins.

These computer generated pictures were inspired by the early cartoon drawings done by Richardson [RICHA 81] representing alpha-helices as cylinders, strands of beta sheets as ribbons with arrows and regions of random coil as lines. Figure 2 · 1 illustrates such drawings where beta sheets are shown as arrows with thickness, helices as spiral ribbons, and nonrepetitive structure as ropes. These diagrams are taken directly from Richardson article on Protein Anatomy [RICHA 81]. The schematic diagrams are useful in showing the overall shapes of protein molecules, relationships between classes and families of proteins, and

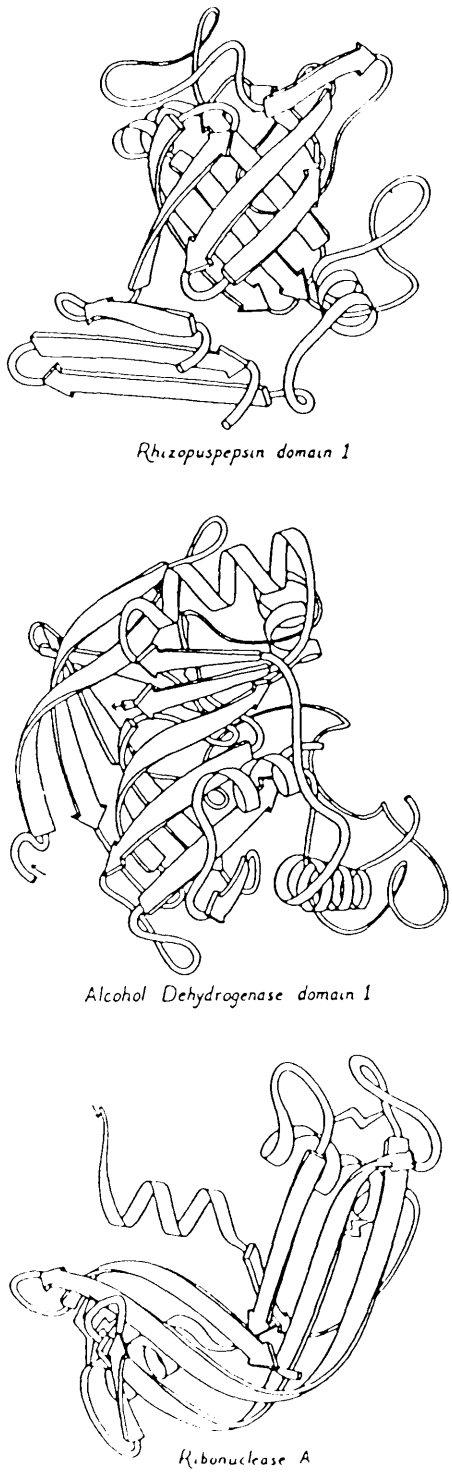


FIGURE 2 - 1 Cartoon Drawings of Proteins.

Cartoon drawings done by Richardson [RICHA 81] representing alpha-helices as spiral ribbons, strands of beta sheets as somewhat flat ribbons with arrows, and regions of random coil as ropes.

topological relationship between elements of secondary and super-secondary structure.

The software requires that secondary structure elements to be included in the drawing are specified by the user. The arrangement and orientation of the alpha helices are made by superimposing the backbone coordinates onto a standard helix oriented along the z-axis; all helical regions are represented by a cylinder with a constant radius, its height is depending on the length of the backbone defining the region. For ribbons, the normal to each peptide group is determined and lines join normals to consecutive groups appropriately. Concerning arrows, the midline of the arrow is computed by a cubic spline fit to the centers of gravity of successive peptides and the orientation of the normal to the arrow is calculated by spline fits to the direction of the normals to the successive peptide groups.

The software handles such techniques as stereo pairs, hidden line removal, surface removal and translucency. Options on the different styles of hidden lines visualization on the vector display are available. The lines can be either removed entirely, changed to dashed lines, or untouched and thus left unchanged. A z-buffer method is used for the elimination of hidden surfaces in the raster display. In this latter display, all the schematic elements are coloured and shaded giving a realistic representation of the model. To specify what portion of the protein is to be labelled is also under the user's control through input data. Manipulation of the model is not possible as the pictures are still.

User communication with the system is almost nonexistent as the only user control of the system is by means of the keyboard.

As this system does not permit interactive manipulation by the user, a group of researchers from IBM UK Scientific Centre have developed a more flexible system discussed in the next section.

2·2·8 I B M UK Scientific Centre

At IBM UK Scientific Centre, in Winchester (G.B.), a large system WINSOM (WINchester SOlid Modeling), was developed ([TODD 83], [BURRI 84] & [QUARE 84]) where the user modifies schematic representations of protein secondary structures interactively. The hardware consists of an IBM 3031 mainframe, coupled to a IBM Series/1 minicomputer which is used as a display controller. This minicomputer controls a calligraphic display (Vector General 3303) and a raster display device (IBM 3279).

The system as a whole is a constructive solid geometry (CSG) modeller used to help model molecules. The CSG modeller is a program which can draw simple geometric primitives such as spheres, cylinders and planes. These can be combined by boolean operations to build more complex shapes. Consequently, very simple application programs are written in order to generate a variety of representations from molecular data. Flexibility of the system comes from the use of a relational database system to collect and structure the data required for a picture ([MORFF 83b] & [MORFF 84]). It can result, for example, in the generation of an appropriately coloured mixture of atom, backbone and ball-and-stick representations [BURRI 89]; electrostatic potential maps may also be displayed. While the major use of WINSOM is to model molecules, it can also be used in archaeological reconstruction, biological applications and sculpture [BURRI 89].

Extension of the project is toward the use of a logic-based language PROLOG, as an alternative to using the relational algebra query system, with the goal of developing a knowledge base system in order to model novel polypeptides and proteins [MORFF 86].

2·2·9 MOLPLOT: Molecule Display using a Microcomputer

MOLPLOT (MOLecular PLOT) has been developed by Clark [CLARK 88] at the Beecham Pharmaceuticals Research Division in Surrey (U.K.). All the programs were written in BASIC using a DEC Professional 300 series microcomputer with a memory of 256 Kbytes. Programs and datafiles are stored either on two floppy diskettes of 400 Kbytes each, or on a 5 to 10 megabyte hard disk. The display device is a monochrome 960x340 bitmapped graphics monitor supplied as standard with the microcomputer.

In spite of the restrictions encountered in memory space, the slow execution process of the microcomputer, the lack of colour and that the fact that BASIC is not an ideal language for molecular graphics programs, the results produced were satisfactory.

The molecular graphics package is well structured into a suite of programs and allows the user to display molecules from Cambridge Crystallographic data files (CSSR) as well as the alpha-carbon backbone of proteins from the Brookhaven PDB data files.

One of the programs is a menu program which displays on the screen the choice of the programs available to the user. Two of the programs deal with displaying and saving small molecules from the CSSR bank and are somewhat irrelevant to our study.

The other programs deal with protein structures by displaying the alpha-carbon backbone of the molecule where alpha-carbon atoms are joined successively. To minimize memory storage, the data are initially preprocessed by allowing only the alpha-carbon entries from the PDB file to be read and stored in a more compact file. This new file also contains some other information discussed further.

Depth cueing of line segments is not done automatically but is available as an option because,for that, a time consuming depth sort is required according to the z-coordinates (effectively the distance from the viewer). To remedy the lack of shading in the system, various line styles --solid, dashed and dotted lines-- are used to indicate the depth cueing effect; the further away the more broken are the lines. Stereoscopic views are available as an option where two views of the molecule are separated by 5 degrees in the y direction and shown side by side. Preselected portions of the protein, which are stored in the input file, can be drawn with different line styles. This variant might be useful if, for instance, a user wants to look at different domains within the same polypeptide chain. The model can be rotated in any of the three cartesian axis by typing in the corresponding code entry for some defined angle displayed as a menu option on the screen. Every time the model is rotated, a rescaling is performed in order to get the whole molecule displayed on the screen. The residues can be labelled specifically with the amino acid residue name and number by typing in the desired three letter code from the twenty existing one. These labels are written in an *"overlay compliment style"* which allows their removal simply by re-writing them with the background colour, namely white. This clever idea sometimes has the side effect of writing over and thus erasing pieces of the backbone chain. Views of the protein can be saved by asking the user to enter a new filename and thus allowing him to resume work the next session. An on-line help manual program is also

available.

The interface with the system is based on a menu command language. The only available input device is the keyboard where the user has to type in an appropriate key code displayed next to each menu selection box.

2·2·10 I H D : Intrasequence Homology Display

The system described in this section shifts a little bit from the above displays of protein structures. It is more related to predicting unknown protein structures by comparing their primary structure; but the way the sequences are displayed is somewhat related to our diagrammatical pictures.

The IHD (Intrasequence Homology Display) system developed by Morris [MORRI 88] at the Physical Chemistry Laboratory in Oxford is based on a Silicon Graphics IRIS 3120 colour workstation with a 1024x768 resolution screen and where up to 4096 colours may be displayed simultaneously. The software has been implemented in FORTRAN 77.

This system is designed for homology modeling. It allows a scientist to detect regions of similarity between two protein sequences and provides him with a variety of tools in the interpretation of sequence homology.

The sequence of the proteins are taken from the National Biomedical Research Foundation's Protein Sequence Database [GEORG 85]. It uses a sequence alignment algorithm [NEEDL 70] that

enables a comparison of all possible segments of a specified length from one sequence with all segments of the same length from the other sequence. A segment is meant to be a group of N successive amino acids in the sequence where N is chosen initially before executing the program. The output of this program gives the highest matching pairwise segments of the two sequences for every segment in the first sequence, along with a score found by the algorithm for this matching pair. A threshold value for the score is fixed and only those matching elements with a higher score are subsequently considered.

It is these results that the IHD presents as a graphical display where the only input device on hand is a mouse. The two sequences are presented as two horizontal lines, one in the upper half of the screen, the other on the lower half, with graduation markers every 10 amino acids and numeric labels every 50 amino acids. Each amino acid is represented as a coloured vertical bar in its appropriate place in its sequence.

Several options are available for the colour coding of the amino acids and have to be chosen before running the program. Depending on what properties --structural, chemical, physicochemical or statistical-- the user wants to concentrate on, a colour is assigned to every group of like amino acids so that every amino acid belongs to a separate colour group. This colour grouping scheme is better than having each amino acid given a different colour as this leads to a confusing display.

The pairwise matching schemes are shown by drawing for every matching pair of sequences, two similar rectangles in each sequence with each one positioned at the location where the matching starts. The height and the width of the rectangle is taken to be respectively the score

and the length of the segment under consideration. Next, a link is drawn for every matching pair from the midpoint of the upper segment to the midpoint of the lower, instead of having a link for every pair of amino acids in the two segments. This is to avoid too much clustering on the screen. From this representation, regions of high sequence homology can be detected as features having tall rectangles linked by connectors. If the connectors are close to the vertical line, this would mean that the homology is meaningful and not just coincidental.

Options are made available to draw a link between each pair of residue that belong to similar regions. Those connecting lines are coloured either as the same colour of the two amino acids if these happen to belong to the same group (called sometimes *"direct hit"*) or otherwise in another colour fixed initially. Another option would be to draw only the direct hits by omitting the other hits. This feature is generally used when the user wants a close-up look at a particular region.

Other facilities are interactive communications with the system that allows the user to ask questions such as the displacement between two similar segments or the length of the predetermined segment. Real-time translation of one sequence in the horizontal axis is possible and helps align particular segments under consideration. Real-time zooming on any region, to detect regions of partial homology is also available.

Another useful utility is the colour editor window which allows interactive changes of the 3 primary colours and permits easy adjustment of these colours for the colour grouping actually in use. The user has only to point at one of the grouping colours shown as a key on the screen and adjust the corresponding colour manually by moving the simulated sliders in the colour editor window. This facility is well suited if, for

instance, each amino acid is assigned a separate group; in this way, the key of one amino acid's colour can be selected and its occurrences can be highlighted in the two sequences by manipulation of the colour editor. Another point worth mentioning is that not too many colours are needed for display; only 16 are sufficient from the 4096 that may be displayed simultaneously out of the 2^{24} available colours available in the palette.

2·3 DIFFICULTIES WITH EXISTING SYSTEMS

From the above review, we notice the wide variety of model representation of proteins. This is understandable because of the fact that the object under consideration has no analogue in real life. Two main streams of representation emerge and depend heavily on the graphics display used. For those systems described based on vector refresh graphics displays, the proteins' models are represented mainly as line segments and the surface of the spheres representing the atoms are approximated by dot surfaces. In these systems, fast interaction and manipulation of the model is effectively performed. However, perception of depth is quite lacking. While intensity depth cueing is a well established technique in these systems, the need for more depth perception is needed. Depth sorting could not be performed for such models as when two lines cross each other there is no way to indicate which one comes in front of the other. To alleviate this problem, some techniques had to be devised. The rocking technique, for instance, allows the model to rock back and forth through a fixed axis. But the inconvenience of this technique is that, if a viewer wants to concentrate on a particular region, as is often the case, no perturbation is allowed and hence the model has to stay still. However, stopping the rocking motion causes the loss of three-dimensional perception.

On the other hand, systems that are based on displaying protein models on a raster graphics display have the advantage of being much less ambiguous in visualizing depth than the models displayed in a vector display. However, they have the disadvantage that manipulation of the model is awkward. For instance, in order to get another view of the model, a user has to wait a couple of minutes and loss of concentration from the user is almost certain.

Thus, in a successful molecular representation system, the model image of a protein should stimulate the imagination of the viewer; the system should allow the user to concentrate on pictures without losing any depth perception of the object displayed, while avoiding a large time delay for the display and manipulation of the images.

Another important feature governing the shape of proteins that needs to be clarified and investigated is the presence of hydrogen bonds. From the systems describing applied methods to visualize secondary structures, it is usually impossible for a user to see through the object displayed. Moreover, we are not able to know how many hydrogen bonds are involved in secondary structure features. Furthermore, there is no way a researcher can investigate unpredictable features such as loop motifs from these models, where all that can be seen are some known specific features. So a factor that should be considered is that it is important not to build in preconceived ideas of what we expect to see, since this will make it harder to discover new features. On the other hand, the well known structures should be clearly visible. This balancing act is another desirable feature that a system should achieve.

Finally, no system readily relates the three-dimensional structure of a protein with respect to its sequence, although this is a major

requirement for predicting protein structure from primary structure. There should be an integrated system where all the displays for three-dimensional structure and for sequence are correlated, allowing the user to follow the relationship.

Chapter 3

PREPROCESSING OF THE PDB

3·1 INTRODUCTION

First, a brief description of the Brookhaven PDB is given in section 3 · 2. The necessary data extracted from it, to be used in the PHD protein visualization package, need to be preprocessed before being stored in various files. Note that another way of storing data would to use a database system. Some protein modeling systems, such as WINSOM [MORFF 83b] and MIDAS [FERRI 88a], are based upon database management systems where both storage space requirements and data access time are minimized. However, in these molecular systems as opposed to ours, huge amount of data is manipulated (e.g. three-dimensional coordinates of proteins, van der Waals surfaces, electrostatic potential fields, etc...) and I believe that the extra complexity of implementing a database system does not give anything special to my system, where data are hold in the form of files and accessed by C

programs.

Upon using the PHD software, three input files are usually required: the first file contains the atomic coordinates of all the atoms in the protein under study (see section 3 · 3); the second file contains the dihedral angles (psi and phi) of all the residues constituting the protein (see section 3 · 4); the third file lists all the hydrogen bonds involved in the protein (see section 3 · 5). A program is described in each of these sections which preprocesses the relevant data. Then, some definitions and criteria for assigning secondary structures from the PDB are discussed (see section 3 · 6). The definitions deal with hydrogen bonds in alpha-helices, beta strands and various turns. They will be needed latter, in Chapter 5, when an algorithm is presented which distinguishes between various types of secondary structure features. Moreover, these definitions and programs are very useful, not only for my work, but can also serve as a starting point for new workers in the field.

3 · 2 THE PDB

The three-dimensional structure of many proteins has been determined by means of X-ray crystallography analysis. Atomic coordinates for many of these proteins are available from the Brookhaven PDB [BERNS 77], where crystallographers deposit their final data in the form of X, Y, Z coordinates for each of the atoms, except hydrogen, of individual molecules. The coordinates of the hydrogen atoms are computed according to criteria explained in section 3 · 5. The accuracy of protein structure assignment is determined by the resolution at which the X-ray crystallography has been performed. For a resolution of 1.5 Å, refined structures are provided where individual side chains can

clearly be detected (but not hydrogen atoms). Other unrefined structures, are also available at a resolution just sufficient to trace the polypeptide chain (resolution generally greater than 2.5 Å). In my research, I work with 54 proteins available through the Biochemistry Department of the University of Glasgow, all at a resolution of less than 2.0 Å.

The entire databank is technically one file with each macromolecule constituting one entry. Each entry has a hundred or so lines at the beginning, giving general information about the macromolecule (name of molecule, name(s) of author(s), comments relating to specific atoms or residues, amino acid sequence, etc...), followed by all the lines (starting with ATOM) giving the coordinates and other information (see below) about each atom. After this come lines (starting with HEMATM) giving the coordinates of any ligands (i.e. small molecules that bind to proteins). At the end of each entry, there are lines that can usually be discarded. The first line of each entry begins with HEADER and can be used to extract the required entry. The complete databank, which is available on magnetic tapes, occupies about 40 megabytes of store.

3·3 ATOMIC COORDINATE FILES

The data provided by the PDB are not in a form most biochemists would find useful [MEYER 74]. Consequently, I have written my own computer program (in C language), named PREPROCESS_COORD, which preprocesses every entry of the PDB, separately. For each atom of each protein, only the relevant data needed for my study of analysis of protein structure are considered, and the following is stored:

i) atom type:		code giving both type of atom and its position within the amino acid e.g. "CA" means alpha-carbon atom
ii) amino acid type:		a 3-letter code, e.g. "ALA" for alanine
iii) amino acid number:		the amino acid number, as found in the PDB.
iv)	X Y Z	} coordinates in 3-D space expressed in Angstrom (1 Å = 0.1 nanometer)

Input to the program PREPROCESS_COORD requires a protein file (an entry from the PDB), say "3TLN", having the format specified by the PDB. The output is a file containing the information listed above (information for one atom per line) starting at the third line. The first line contains the name of the macromolecule, followed in the second line by the total number of atoms in the protein and the number of residues; these numbers are needed as memory is allocated dynamically for such input data in my PHD software program (see Chapter 5). The base name of the output file is the same as that of the input file but with an extension ".mol". For example, if the name of the input file is "3TLN", the name of the output file will be "3TLN.mol". Figure 3 · 1 displays a portion of the "3TLN.mol" file.

```
3TLN: THERMOLYSIN
NUM_ATOM= 2432  NUM_CA= 316

N    ILE   1    34.321    40.417    -.972
CA   ILE   1    33.448    41.147    -.013
C    ILE   1    33.896    42.574     .197
O    ILE   1    35.104    42.932     .315
CB   ILE   1    33.309    40.228    1.248
CG1  ILE   1    31.792    39.964    1.498
CG2  ILE   1    34.002    40.619    2.562
CD1  ILE   1    31.498    38.429    1.577
N    THR   2    32.934    43.487     .355
.  .  .  .  .  .  .  .  .  .  .  .  .  .  .
.
```

FIGURE 3 · 1 Preprocessed Coordinate File "3TLN.mol"

3 · 4 DIHEDRAL ANGLE FILES

PREPROCESS_DIHED is a C program which generates all the dihedral angles of a particular protein by storing in a file the psi and phi angles for each amino acid. Figure 3 · 2 shows a portion of such file for the Thermolysin protein; the first line contains the name of the protein. Starting at line two, dihedral angles are listed by specifying the one-letter code of the residue (see Appendix 1), followed by the psi and phi angles associated with the residue. The base name of the output file is the same as that of the input file, but with the extension ".dihed". The input file should be of similar format as the preprocessed coordinate file.

```
3TLN: THERMOLYSIN

I          143          0
T          125        - 98
G         -166        -171
T          128        - 94
S          135        - 71
T         -178        -145
V          130        -130

.  .  .  .  .  .  .  .  .
```

FIGURE 3 · 2 Preprocessed Dihedral Angle File "3TLN.dihed"

3 · 5 HYDROGEN BOND FILES

3 · 5 · 1 Placement of Hydrogen Atoms

Hydrogen bonding is of great importance in chemistry and biology. Three different type of bonds are categorized : inter-mainchain, sidechain — mainchain, and inter-sidechain hydrogen bonds. These hydrogen bonds consist of a sequence O····HN or OH····O, joining different amino acids in the chain. One problem is that the position of hydrogen atoms is not known from the PDB, due to limitations in the resolution of X-ray crystallography.

In my work, the position of all the relevant hydrogen atoms are calculated by the procedure similar to the one used by Baker and Hubbard [BAKER 84]. The placement of a hydrogen atom depends on whether it is located on the main chain or the side chain. For example, in inter-

mainchain hydrogen bonds, only main chain amide hydrogen atoms (hydrogen atom bonded to the main chain nitrogen atom) are considered. For these bonds, the hydrogen positions are generally known from standard geometry : all hydrogen atoms bonded to their corresponding nitrogen atom have a standard NH bond length of 1 Å; the coordinates of the hydrogen atom are computed by placing the hydrogen atom on the bisector of the angle C · N · Cα , and in the plane defined by C, O, N (see Figure 3 · 3).

The position of the relevant hydrogen atoms on the side chains (for sidechain — mainchain and inter-sidechain hydrogen bonding) are computed in a slightly different manner (see Baker & Hubbard article [BAKER 84] for further details).

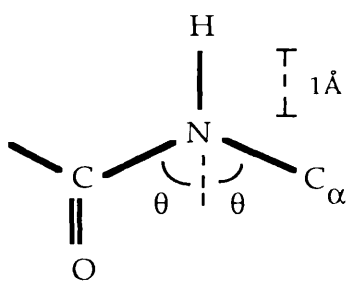


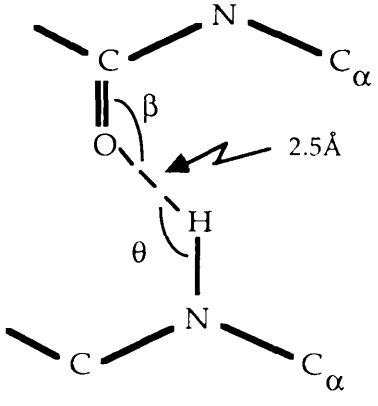
FIGURE 3 · 3 Hydrogen Atom Placement on the Main Chain

3 · 5 · 2 Hydrogen Bond Computation

A major problem arises when a user wants to define what constitutes a hydrogen bond. In other words, there exists no universal definition of hydrogen bonds. Kabsch & Sander [KABSC 83] define it by using electrostatic energy; others consider criteria about only distances

between specific atoms (usually PDB crystallographers' data). In my research, I use Baker and Hubbard rules [BAKER 84] to define hydrogen bonds; they are simple and effective.

Once the coordinates of the relevant hydrogen atoms are computed, hydrogen bonds are found by applying the Baker & Hubbard criteria. For instance, the inter-mainchain hydrogen bonds are found in the following way : all combinations of O...H (where O belongs to the main chain CO group and H to the amide group NH) which satisfy the correct angles (i.e. $N \cdot H \cdot O > 120°$ and $H \cdot O \cdot C > 90°$) and distance rules (i.e. O...H < 2.5 Å) are accepted as being inter-mainchain hydrogen bonds joining the NH to the CO group (see Figure 3 · 4). Disulphide bonds are included as being inter-sidechain hydrogen bonds; a pair of cysteine residues forms a disulphide bond if the distance separating the two sulphur atoms is less than 1.5 Å. The other inter-sidechain and sidechain — mainchain hydrogen bonds are found according to other rules defined in Baker & Hubbard review.



Baker & Hubard criteria:

- $\beta > 90°$
- $\theta > 120$
- O···H < 2.5 Å

FIGURE 3 · 4 Conditions for Acceptance of Inter-Mainchain Hydrogen Bonds

The program PREPROCESS_HBOND, written also in C language, generates all the hydrogen bonds of a particular protein by finding all possible O...N, O...O and S...S pairs; computing the relevant hydrogen atom coordinates; and applying tests of distances and angles. This is done in an exhaustive manner from the N-terminal to the C-terminal amino acid of a protein. This program was implemented by a collaboration between Dr. Poet and myself.

The program also requires the same input file as the one used in PREPROCESS_COORD. The output is a file listing all the hydrogen bonds (one hydrogen bond per line) starting at the third line. The first line contains the name of the macromolecule, followed in the second line by, successively, the total number of hydrogen bonds in the whole protein, the inter-mainchain, the sidechain — mainchain and the inter-mainchain number of hydrogen bonds. The base name of the output file is the same as that of the input file but with the extension ".hbond". Figure 3 · 5 shows a portion of the "3TLN.hbond" file. Starting at line three, hydrogen bonds are listed with the following information:

i) *Group 1 - hydrogen bond category* -

It is a string of 3 characters, where the first characters represent the category type of hydrogen bonds (e.g. '4' : inter-mainchain; '5' : sidechain — mainchain; '6' : inter-sidechain). The last two characters represent the two connectors of the hydrogen bond (e.g. 'N' : NH group; 'O' : CO group; ' S' : Sulfur, etc...). As an example, '4NO' stands for inter-mainchain hydrogen bonding between NH and CO groups).

ii) *Group 2 - Length information -*

- 'length' of the hydrogen bond computed as the absolute value of the difference, R, between the PBD amino acid numbers of the two residue involved.

iii) *Group 3 - Information on first group -*

- PDB amino acid number
- one-letter amino acid code identifier
- atom number reference, n, involved (e.g. n = -1 for N; n = 0 for CA; n = 1 for C; n = 2 for O; n > 2 for side chain atoms)

iv) Group 4 - Information on second group -

- PDB amino acid number
- one-letter amino acid code identifier
- atom number reference involved

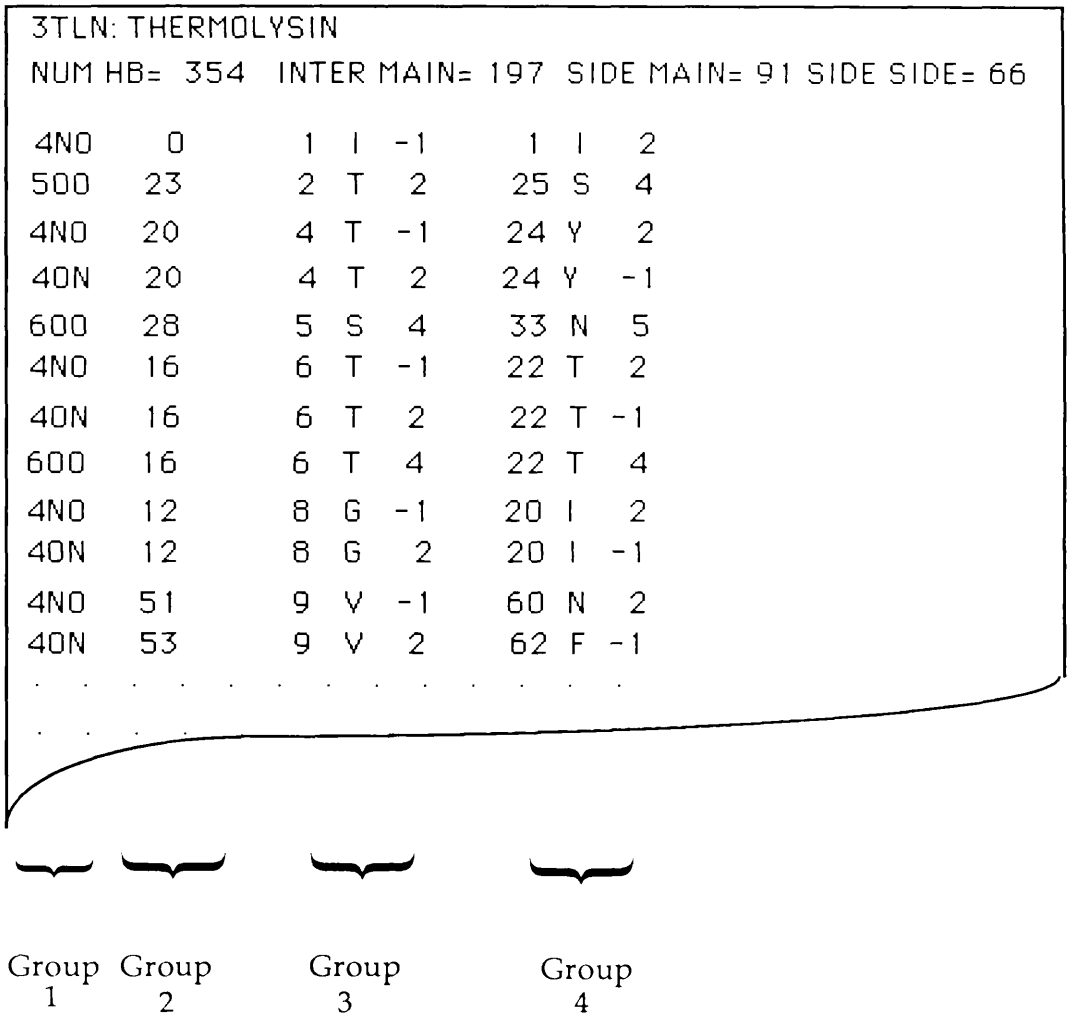


FIGURE 3 · 5 Preprocessed Hydrogen Bond File "3TLN.hbond"

Note that for all hydrogen bond entries, the PDB amino acid number of group 3 is never greater than the PDB amino acid number of group 4; this is to force the orientation of a hydrogen bond to go from the N-terminus to the C-terminus. These hydrogen bonds are listed in ascending order (with respect to their group 3 amino acid) from the N-terminus to the C-terminus. When more than two bonds are involved at a common amino acid, the inter-mainchain bonds are listed first then

come the sidechain — mainchain bonds and finally the inter sidechain ones.

3 · 6 INTER-MAINCHAIN HYDROGEN BOND PATTERN DEFINITIONS

The successful analysis of the relation between the sequence and the structure of a protein requires an unambiguous and physically meaningful definition of secondary structure. In the Brookhaven PDB, several secondary structures have been assigned by crystallographers. However, several shortcomings have been found in the existing compilations due to the fact that these assignments are often subjectively defined and sometimes incomplete [KABSC 83].

There has been no standard definition of what constitutes a secondary structure. Several algorithms are found throughout the literature that extract structural features from the atomic coordinates (for review see [ROSE 76], [LIFSO 80], [LEVIT 77] & [KABSC 83]). Each one of these approaches has its strengths and weaknesses and depends on the purpose for which the research is aimed. The definitions I use are taken from Kabsch and Sander [KABSC 83] where the authors develop a pattern recognition process for hydrogen bonds in secondary structures. In this section, first, some elementary hydrogen bond features such as "turns" and "bridges" are defined, and, based on them, α-helices and β-ladders. In other words, secondary structures are recognized as being repeats of elementary units of hydrogen bonding patterns.

3 · 6 · 1 Elementary Hydrogen Bond Patterns

The elementary hydrogen bond patterns are of two sorts: turns and bridges. The turn pattern is a single hydrogen bond of type (i, i + n), often called an *n-turn* (with 1< n <6), that exists between the CO group of amino acid i and the NH group of amino acid (i + n). This can be annoted as : n-turn(i) = Hbond(i,i+n), n = 2, 3, 4, 5.

Figure 3 · 6 illustrates a 3-turn hydrogen bond.

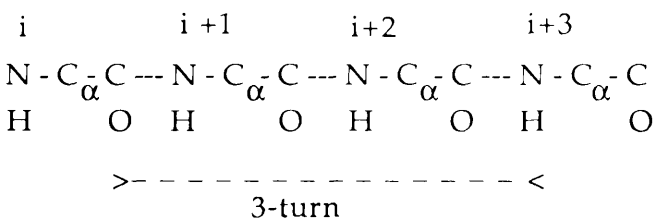


FIGURE 3 · 6 A 3-Turn Hydrogen Bond

When two non overlapping stretches of three residues each, say (i – 1, i, i + 1) and (j – 1, j, j + 1), form a bridge, being either a parallel or antiparallel one, depending on the pattern of the bridge. Figure 3 · 7 illustrates these two notions. Note that each hydrogen bond in this pattern has n, the difference between the two amino acid sequence numbers involved in the hydrogen bond, greater than 5.

3 · 6 · 2 Patterns for Alpha Helices

Repeating *turns* are helices. We usually find three types of alpha

helix according to the type of turns found i.e. alpha_3, alpha_4 and alpha_5. A minimum helix is defined by two consecutive n-turns. The alpha_4, generally called an alpha helix, is the most common and is defined by a series of successive 4-turns. Alpha helices can have some irregularities, where not all hydrogen bonds are formed, and thus make them imperfect.

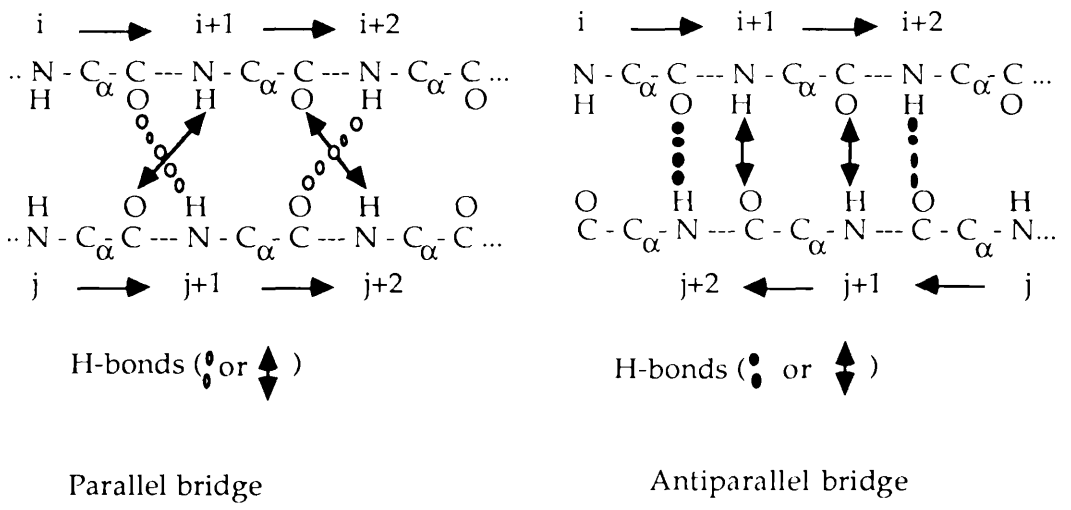


FIGURE 3 · 7 Bridge Patterns

3 · 6 · 3 Patterns for Beta Sheets

Repeating bridges are *ladders* (or *pair of strands*) where a ladder is defined as a set of one or more consecutive bridges of identical type. Connected ladders are *sheets*, either parallel or antiparallel, where a sheet is defined as a set of one or more ladders connected by shared residues. A somewhat more flexible definition of sheets is used throughout my research. In the rules governing sheet bonds, the same definition of a ladder applies with some additional flexibility : e.g. a

missing bond from a bridge is permissible but no two consecutive bonds from two consecutive bridges should be missing; in such a case, the ladder is divided. Figure 3 · 8 illustrates two cases. In Figure 3 · 8 · a, only one bond is missing from the second bridge and thus one ladder is considered. In Figure 3 · 8 · b, two consecutive bonds from two consecutive bridges (bridge 2 and 3) are missing; in this case the ladder is divided into two ladders. The algorithm which detects such features is described in Chapter 5.

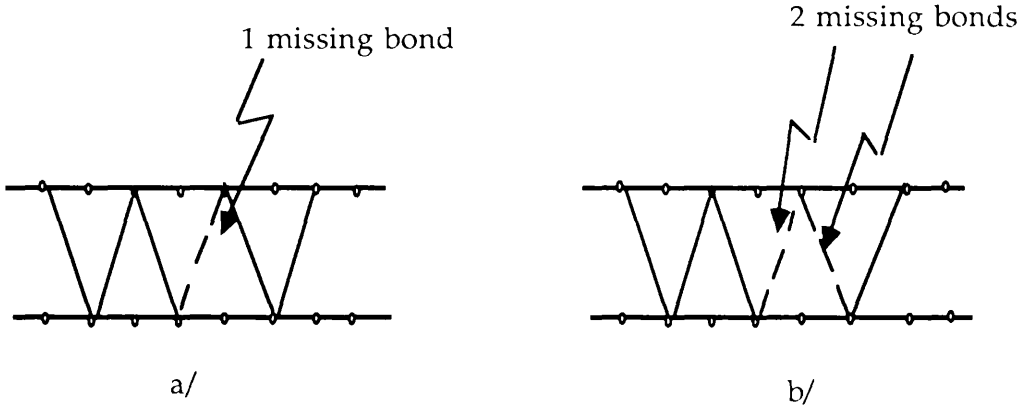


FIGURE 3 · 8 Formation of Ladders

3 · 6 · 4 Patterns for Loops and Various Turns

Various sorts of nonrepetitive structures have also been defined in the literature. Milner-White and Poet [MILNE 87b] describe hydrogen bond patterns for various nonrepetitive structures such as gamma-turns, beta-turns, beta-hairpins, paperclips, beta-bulges, some other loop motifs, as well as for alpha helices and beta sheets.

For instance, a gamma turn is a 2-turn hydrogen bond; a beta-turn is

defined as being a 3-turn hydrogen bond which does not belong to an alpha helix; a paperclip is a loop motif commonly found at the C-terminal ends of alpha-helices and can recognized as a succession of a 5-turn and a 3-turn hydrogen bond; a beta-bulge can be regarded as occurring where an extra amino acid is inserted in one ladder, or strand, of beta-sheet , thus disrupting the hydrogen bonding and causing a bulge in the beta-sheet. These definitions will be used throughout the thesis.

Chapter 4

PROTEIN VISUALIZATION AND USER MANIPULATION

4·1 INTRODUCTION

In this chapter, the user interface design and the user manipulation of the PHD protein visualization software package are thoroughly explored and described, leaving all the details of internal structure, data structure and manipulation, and algorithm descriptions, until the next chapter.

4·2 TASK ANALYSIS

The first step was to define the task analysis of the project. This initial stage must always preceed the design of an interactive system where user's needs are carefully studied. It often involves interviewing prospective users, studying their working environment and measuring

their performance [NEWNA 79]. A synthesis of this analysis forms the starting point for the user interface design.

The task analysis of my system has been carried out with a close collaboration of the Biochemistry Department at the University of Glasgow. Dr. Milner-White, the end user and an expert in protein structure analysis, has devoted considerable time to define and study the goals of the system.

The goal of the research was to develop ways of picturing proteins that assist in understanding both their three-dimensional structure and the relationship between it and their sequence.

On one hand, the system must allow the user to direct the effort toward his investigation, and not toward the system. For that, a transparent view of the system is required, avoiding the user, as far as possible, knowing anything about the internal operations (i.e. knowledge of file structure, internal data structure, etc...). This is particularly important in systems to be used by researchers who do not have a strong computer background.

On the other hand, another fundamental question arises as to what information should be presented on the screen and how to display it in the most effective manner. In order to understand the three-dimensional folding of these extremely complex objects, a facility for examining them at several levels of their structure is essential. The main levels of proteins, selected to bring out the most valuable insights in the project under study are : overall shape, chain structure and folding, secondary structure, atomic details and sequence. It is difficult to display all such information in one picture, for these very large molecules,

because of the huge amount of detail involved. The approach taken in the present work concentrates on developing relatively simple and informative pictures that focus on the relevant features appropriate at the different levels. The strategy adopted in picturing proteins is based on the idea of multi-level displays, where the pictures can be manipulated interactively and displayed simultaneously, taking advantage of window management techniques incorporated in window systems; in such environment, interesting sections of proteins can be selected, showing chain structure, atomic details or diagrammatic views.

However, no standard window system has been adopted by the various vendors in the industry trend to make software portable from one machine to another. There exists two types of window systems. The so called kernel-based window systems (such as the Apple Macintosh Toolbox), closely tied to their respective hardware, are difficult to port to other machines; in these environment, window system functions are handled the same way as other operating system calls [STERN 87]. The other recent window systems encountered are the network-based window systems (such as the X Window System), where the window system is not part of the kernel. For example, the X Window System is based on a client-server model. The server process (or display server), attached to each physical display on a network system, controls the display screen, keyboard and mouse of its corresponding workstation. All device-dependencies are encapsulated by the server while client applications are device-independent. In other words, once a server has been implemented for a machine, any application using the X Window System can be brought to the machine. Therefore, a programmer who wishes to write applications that run under X Window System must perform all graphics routines by using only procedures from the X graphics library available (called Xlib). Writing an application this way makes it portable to any hardware supporting an X server by simply recompiling it; no

changes in the code are required ([SCHEI 86] & [STERN 87]). From this analysis, it is clear that an X Window System is suitable for my application in order to make it portable to various hardware equipments. However, this has not been the case, mainly, because X (version 11) was not installed until May 89 in our Computing Science Department. Consequently, due to lack of time, it was not possible to rewrite the whole application in an X environment. It is currently implemented on a Whitechapel colour workstation which supports a kernel-based window system.

Going back to the application in itself, the user's ability to visualize a three-dimensional structure from a two-dimensional display is also an important factor to consider and is related to communication between the user and the system. This point should be considered in displaying polypeptide chain structures, as well as their atomic detail elements. Another prerequisite for these representations is to be able to see through the object displayed. Moreover, a way to correlate the relationship between three-dimensional structure of protein and sequence (or primary structure) should also be displayed. An aspect that should be borne in mind is to maintain a consistent format from one display, say the polypeptide chain picture, to another, say the primary structure view.

The study of a biochemist working environment shows that a need for more information detail is needed to detect protein structure features. When pictures of whole proteins are displayed in the currently available systems, they are almost invariably represented as chains with the alpha-helices and strands of beta-sheet singled out in some way (see Chapter 2). Although this is a convenient representation, it does make the assumption that these particular features are the most important ones. There is no way a researcher can investigate unpredictable features from

these models. So a factor that should be considered is that it is important not to build in preconceived ideas of what we expect to see, since this will make it harder to discover new features. On the other hand, the well known structures should be clearly visible.

This philosophy has guided the development of the user interface of the PHD protein visualization system and formed the basis for deciding how the package should be presented to the user. Shneiderman's "Golden Rules of Dialog Design" [SHNEI 87] have been closely followed as a guidance for the design of the menu driven system. Whenever possible and appropriate, these rules were applied.

4·3 WINDOW ARCHITECTURE

4·3·1 CG1 Window Manager

The CG1 Window System provides the user with interactive controls over the position and size of each window displayed on the screen. These controls are part of the Window Manager. They are an integral part of the GENIX[1] programming environment and of any application developed for the CG1 using the Window System. The user should be aware of the following points whenever using an application. He can change any window's size and position. He can also change a window priority relative to other windows as the CG1 supports overlapped window models. Moreover, he can stow a window in its icon, or unstow the window to display it back on the screen. For more information about the interface with the Window System incorporated in the CG1, the reader is recommended to read the user manual of the

---

1 The CG1 runs under GENIX, a variant of the UNIX Operating system [GENIX 84]

Whitechapel [WHITE 84] and the methodology of window management book edited by Hopgood and al. [HOPGO 86].

For the PHD application software, the user has to be aware of the conventions listed below, which are determined by the Window Manager. The windows displayed follow the following rules:

- Corners have a fixed appearance, and contain controls for moving, changing size and changing priority.

- Icons are displayed as squares with a title name of the corresponding window, and have the lowest priority.

A typical window is shown in Figure 4 · 1, where the text in the title bar is only accessible by the application program.

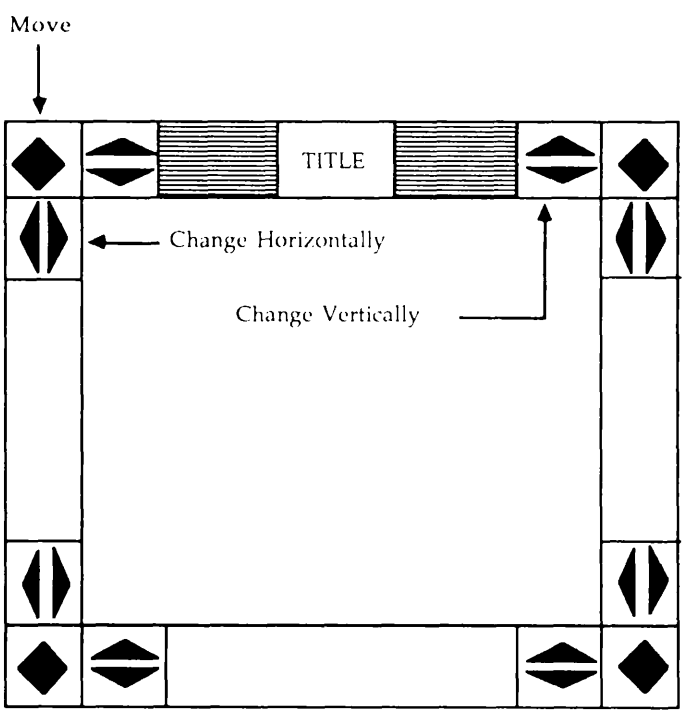


FIGURE 4 . 1 A Typical Window in the CG1

4 · 3 · 2 The PHD Window Architecture

In the PHD software application, there are 5 different windows:

- *Polypeptide Chain Window*
- *Atomic Detail Window*
- *Diagrammatic Window*
- *Message Window*
- *Colour Key Window*

As a convention, I will be referring to these windows throughout the rest of the thesis by writing them in italic font. The overall conceptual image of this two level window system is shown in Figure 4 · 2. The *Polypeptide Chain Window* is the root window, or parent window, where the polypeptide chain is displayed. The two following windows display, respectively, atomic details and primary structure diagram pictures, of the protein under investigation. I refer to these primary structure displays as diagrammatic views. These two windows, the *Atomic Detail Window* and the *Diagrammatic Window*, are called children windows of the root window. Access from the parent window to one of the children windows is related to the state of the root window. For instance, in obtaining a diagrammatic view of a protein from its polypeptide chain, the view displayed on the *Diagrammatic Window* initially is the one representing exactly the portion of the chain displayed on the parent window at the time of the transfer. As another example, the atomic detail picture of a segment of the chain is displayed initially on the *Atomic Detail Window*,with the same orientation as the polypeptide chain at the time of the transfer. This avoids disrupting a user's train of thought. In both parent and the children windows, either the inter-mainchain or the sidechain — mainchain hydrogen bonds can

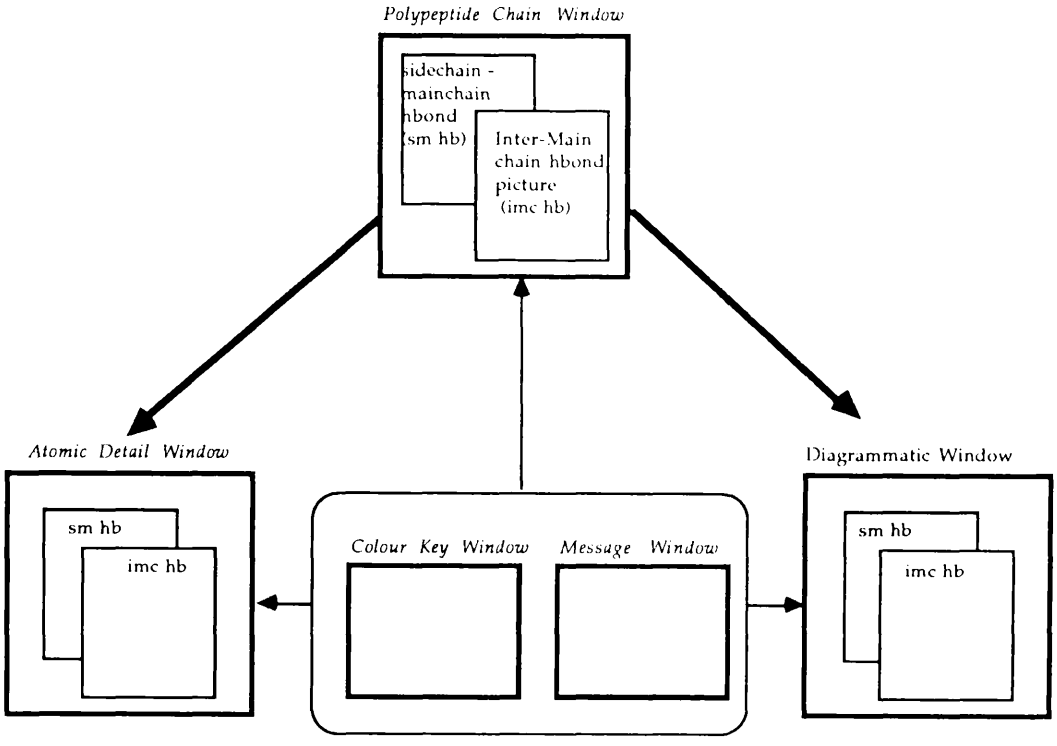


FIGURE 4 · 2 The PHD Window Architecture

be displayed but not both at the same time. This is to avoid a cluttered appearance and the display of too much detail, that might confuse the user.

The *Message Window* is used for error messages, prompts and feedback from the system. The last window, the *Colour Key Window*, is used as a colour editor display of the hydrogen bonds. It displays the different colours allocated to the hydrogen bonds in the picture actually seen, either on the parent or the children windows. It also allows the user to design interactively his own colour scheme for hydrogen bonds.

All the windows have a title name referring to their respective window name, except for the parent window which has, in addition, the name of the protein under study. An instance of the position of the different windows of the system is illustrated in Figure 4 · 3, during a session with the thermolysin protein (coded as 3TLN); the picture is taken directly from the screen of the CG1 workstation.

4 · 4 MENU DESIGN

4 · 4 · 1 Menu-Based Interface

While command-driven language systems require instructions memorization from the user which can vary from one system to the other (as no universal set of instructions have the same meaning to all people [LANDA 83]), and that frequency of errors in commands and data entries may discourage users from effective use of such systems, menu systems on the other hand, provide a simpler and more structured approach to the Human Computer Interface aspect. As Allen pointed

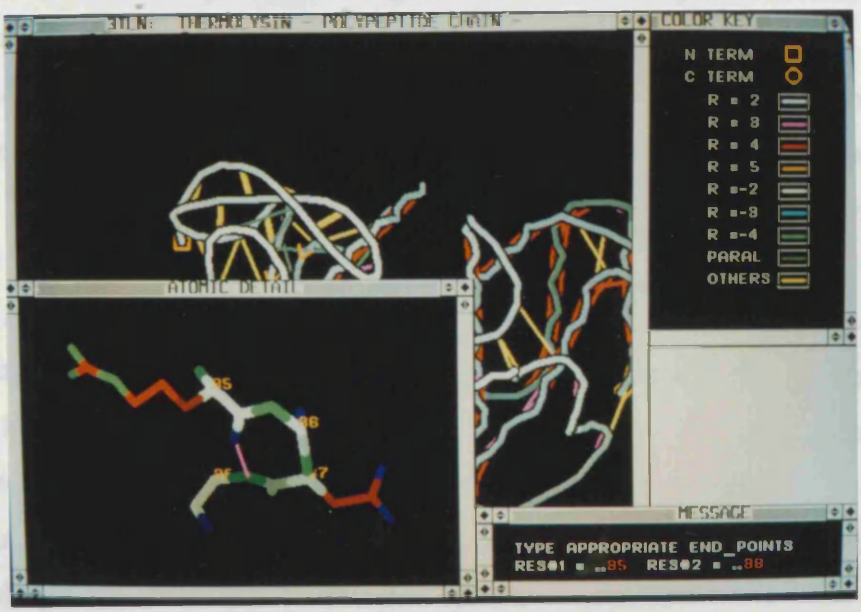


FIGURE 4 · 3 An Instance of the Position of the Different Windows During a Session With the Thermolysin Protein (3TLN)

Four windows are displayed. In the *Polypeptide Chain Window*, the inter-mainchain hydrogen bonds in the three-dimensional structure of 3TLN are displayed. The *Colour Key Window*, appearing at the top right corner of the picture, displays the different colours allocated to the hydrogen bonds actually seen in the polypeptide chain image; this window is also used as a colour editor where the user can design his own colour scheme for hydrogen bonds. In the *Message Window*, located at the bottom right corner, two numbers were entered by the user to get a closer view of the protein in the region between amino acid 35 and 38. On the bottom left corner, the *Atomic Detail Window* shows the atomic details of the chosen segment with the hydrogen bonds involved.

out [ALLEN 82], menus incorporate the desirable cognitive feature of recognition, rather than recall, of the item or command needed. In this environment, the user does not have to memorize input options since a range of options at a particular stage is listed. It also prevents him from choosing invalid options. Despite the fact that some arguments have been made that menus can reduce the execution speed of a system, and that there is evidence that slow execution may be directly related to worker dissatisfaction [BARBE 83], menus have been used to provide a quicker and more pleasant learning experience for novice computer users (as this is the case with the end users of my product) [HODGS 85]. All of the arguments listed above made me adopt the logical solution of a menu driven interface for my system. Moreover, it also fulfills one of the objectives set by Dr Milner-White (the end-user), which is to gain immediate use and productivity on the system for it to be successfully accepted by the biochemistry community.

4 · 4 · 2 Choice Devices

Generally, an interactive graphics application program defines input data to the program using 6 classes of logical input devices: *locator, stroke, valuator, choice, pick and string*. These logical devices are frequently generated by physical devices associated with graphics displays and can be conveniently echoed on these graphics displays [HOPGO 83]. The logical *choice* input class is defined as one that enters a selection from a set of alternatives. It is usually used as a basis of a menu selection tool [SALMO 87]. The actual realization of the *choice* device on a physical device can vary quite significantly, from a menu hit by a lightpen to a name typed at the keyboard. Menu selection using a mouse seems an adequate solution for our system since this hardware tool is a fast and

accurate pointing device, and furthermore, it is available on the CG1. In this environment, the user makes a selection by positioning the cursor displayed on the screen at the menu position, then clicking the mouse to confirm his choice and observing the resulting effect.

4 · 4 · 3 Menu Architecture

Communication with the system is thus accomplished by the selection of a command from menus attached to the parent and children windows. Each window has a specific menu associated with it. In order to avoid user confusion, the menu architecture is designed as a multilevel structure having a maximum of three levels (or depth). Hence, a selection from a parent menu might bring up a child menu at the second level, which in turn can bring up a last menu. Travel from the parent to one of the child windows and vice versa is possible through menu selection.

4 · 4 · 4 Information Display and Feedback

The number of items in a particular menu seldom exceeds eight. This avoids time delays in user selection and is economical with space on the screen. Item selections are presented as character strings. Their terminology and meaning is simple and is appropriate for a biochemist's environment.

Feedback from user selection is always supplied by the system to avoid user frustration. When a choice is selected, three events occur : the pixels in the menu item box are inverted from white to black; a star

(*) is printed next to the corresponding character string; and the iconic shape of the cursor changes from an arrow icon to a cross icon. Once the operation finishes, the cursor regains its original arrow iconic shape. This is to tell the user that the system is processing the operation requested and avoids him being disconcerted. All these supplied feedbacks are particularly important as the user does not have to wonder about what the system is doing, or whether the selection should be given again in the case of a relatively high response time. In the PHD system, this maximum response time is 5 to 6 seconds and is caused by the transfer from the parent to the *Diagrammatic View Window* when a diagrammatic view is displayed for the first time. All the other selections take a shorter time (in the order of 1 to 3 seconds) to be performed.

Finally, all the menus are always displayed at the top right hand corner of the screen. This constraint is not seen as a weakness of the system but, on the contrary, helps the user to become used to making each selection in a fixed location [HEARN 86]. Before starting a session, a user has the option to change all menu positions as well as window position and sizes, without recompiling the whole program.

Some of the most common menu commands are described in the course of this chapter. The close relationship between the different pictures facilitated consistency in designing menu entries in the different pictures. Several menu commands performed at the parent window level have similar effects (and therefore have the same header name), when performed in the other two child windows. While the actions of the different commands in the *Polypeptide Chain Window* are explained thoroughly, similar ones found in the other pictures will be systematically listed without too much detail. As another convention, in order to avoid confusing the user between windows and menus, all menu commands will be written in capital letters.

4 · 5 POLYPEPTIDE CHAIN WINDOW

4 · 5 · 1 Path of the Polypeptide Chain

Pictures that display polypeptide chains by drawing lines joining successive alpha-carbon atoms and discarding all other atoms are widely used (see Chapter 2). They are often called alpha-carbon plots and sometimes referred to as virtual bond plots. Such pictures are often dominated by small scale oscillations from one atom to the next and this makes it hard to follow the path of a chain, especially for a large protein.

The course of the chain may be further clarified by means of an averaging procedure that was developed by Milner-White and Poet ([MILNE 85] & [POET 86]). Such plots are especially useful for displaying features of the whole large proteins while allowing a user to see through the object displayed. The line segments have various thicknesses and colour coding, which help represent features of the polypeptide chain. However, the original pictures of the project could not be manipulated interactively. This was partly due to the implementation of a time consuming procedure drawing 'tramlines' around line segments, which gives a three-dimensional depth effect to the whole picture. Henceforth, some other implementation techniques have to be devised in order to achieve interactive manipulation of pictures while not losing depth perception. The technicalities of these techniques are explained in Chapter 5.

In my software package, the smoothing procedure may be carried out by using any odd number of alpha-carbon atoms for averaging. It replaces the exact atomic coordinates of the alpha-carbons with the

average over several neighbouring ones. For example, an averaging factor of seven means that we sum each of the coordinates of the atoms in question together with three atoms on each side and divide by seven. Two special cases are at the N-terminal and C-terminal regions of the chain where, respectively, the three anterior and the three successor alpha-carbons do not exist. In these cases, the average is taken over the alpha-carbon in question and the six following or preceding ones depending on whether the location of the alpha carbon is at the N-terminal or C-terminal region of the chain.

Milner-White and Poet found that an averaging factor of seven produces especially smooth lines because of the fact that alpha-helices have about 3.6 amino-acids per turn [PAULI 51], so that seven amino acids wind round the helix almost exactly twice. Plots with values of N (where N is the averaging factor) of seven, or even fifteen, have been found to be useful especially for very large proteins because the details of alpha-helices, as well as several other features, are smoothed out. They avoid the visual confusion due to sharp bends and zig-zags so that the overall course of the chain may be followed unambiguously. They also help to display domains and subunit structure of large proteins. However, in the majority of instances, an averaging factor of five is better, since oscillations remain that are useful for indicating alpha-helices.

This averaging curve, which is a simple algorithm compared to others such as the cubic-spline approximations, always stays close to the original curve. Some errors occur when a bend is found and the curve will always appear on the inside of the bend causing sharp corners to be cut slightly [POET 86]. However, the exact shape of these turns can be visualized by displaying their atomic detail features in the *Atomic Detail Window*, as explained in the appropriate section (see section 4 · 6).

The system allows the user to work with any averaging factor. Selection of an averaging factor is carried out by first selecting the AVERAG FACT entry from the Polypeptide_Chain menu. The Average_Fact submenu appears on top of the main menu. Six predefined options are available from this submenu; the first five correspond to the most effective averaging factors. The options are: 1, 3, 5, 7, 15 and OTHER. Note that the value of 1 displays the chain without the averaging technique, and thus shows the crude alpha-carbon plot. It is useful when a researcher looks for short turn inter-mainchain hydrogen bond patterns of less than three amino acids apart; these patterns are sometimes hidden by the chain when $N \geq 3$. The OTHER option is used in case the user chooses his own factor. In this case, the *Message Window* appears and echoes for the number to be entered.

Since the averaging method works only with odd factors, entry of even numbers is automatically incremented by one. If the factor N happens to be out of range (i.e. $N < 1$ or $N > m$, m being the number of alpha-carbons in the chain), a correction is made by choosing the closest odd number to either one of the chain extremities.

4 · 5 · 2 Hydrogen Bond Patterns

To describe the overall structure of a protein or domain, it is usual to give the number of alpha-helices and beta-sheets it contains. However, in proteins that have been studied by X-ray crystallography at high resolution, far more information can be extracted at this level of structure. Firstly, there are often differences in hydrogen bonding between helices or strands, especially at the ends. Secondly, several common non-secondary structure features have been identified, that may

be collectively called loop motifs, on the basis of their patterns of inter-mainchain hydrogen bonds ([MILNE 87a] & [MILNE 87b]).

A substantial proportion of hydrogen bonded interactions are provided by side chain atoms. These provide important stabilizing interactions at the protein surface and sometimes in the molecular interior, as mentioned by Baker and Hubbard [BAKER 84]. It was noticed that more main chain CO groups than NH groups of the main chain are hydrogen bonded to side chain atoms, and that local interaction ($R < 4$) involving such bonds is very frequent. Also many of the sidechain — NH hydrogen bonds are found to be at the N-termini of alpha-helices.

A decision has been made to display separately the two different types of hydrogen bonds (inter-mainchain and sidechain — mainchain) in all the pictures. This avoids a cluttered display with excessive amount of information which may lead to confusion, and thus violate some of the criteria fixed at the start of the project.

Hence, depending on the category of hydrogen bonds to be visualized with the polypeptide chain, two options are possible: either the inter-mainchain bonds are displayed (choosing the MAINCHAIN HB option), or those involving sidechain — mainchain bonds (SIDECHAIN HB option).

All the hydrogen bonds are sorted and plotted in depth order, along with the chain line segments, to give an impression of depth. They are, however, offset in the z-direction by a small amount (4 Å), to avoid clutter when displaying, for instance, alpha-helices which contain many hydrogen bonds and chain line segments. The hydrogen bonds are also colour coded, but the colours indicate the type of bonds rather than depth. No tramlines are used for such hydrogen bonds as they might hide some

part of the chain.

4 · 5 · 2 · 1 Inter-Mainchain Hydrogen Bonds

If the inter-mainchain hydrogen bonds are displayed (MAINCHAIN HB option), a variety of colours is used for these bonds depending on R, the difference between the amino acid sequence numbers of the NH and CO groups involved. These hydrogen bonds are portrayed as lines joining the relevant 'smoothed' (using the averaging technique) alpha-carbon atoms. Here, the colours represent the distance apart in sequence of two amino acids joined by a bond. Also, whenever there is a pair of hydrogen bonds between both sets of CO and NH groups of two amino-acids (a feature found usually in antiparallel beta sheets), an extra thick line is drawn. The colour allocated to such lines is that of the bond with the negative value of R. The legend for the colour coding of hydrogen bonds and the reallocation of their colours by the user is explained in section 4 · 9, describing the *Colour Key Window*.

In the polypeptide chain pictures, different chain structures produce distinctive hydrogen bond patterns. Alpha-helices are easy to recognize as they appear as wavy lines, mingled with roughly parallel hydrogen bonds that indicate special features of each helix. Parallel beta-sheets show up as a characterized zig-zag pattern of bonds between parallel parts of the chain, whereas antiparallel beta-sheets produce a ladder pattern, with thick bonds forming the rungs of the ladder. While both alpha-helices and beta-sheets are characterized by regular, repetitive patterns of hydrogen bonding, some nonrepetitive elements of structure, known as turns, can also be displayed. They are easily visualized by the colour and pattern of the bonds. The N-terminal and C-terminal amino acids are

easily picked out by coloured symbols in the polypeptide chain pictures. Their colour code is also displayed in the *Colour Key Window* (section 4 · 9).

Figure 4 · 6 illustrates the inter-mainchain hydrogen bonds involved in 3TLN. The original protein coordinates are from X-ray crystallography work by Holmes and Matthews [HOLME 82]. The folding of the protein can be seen easily from the polypeptide chain, which is drawn as thick white or gray lines joining successive smoothed alpha-carbons. Inter-mainchain hydrogen bonds are drawn as coloured lines joining the positions of appropriate main chain atoms. Alpha-helices appear as wavy lines, with red hydrogen bonds; hydrogen bonds in antiparallel beta sheets are portrayed as thick yellow lines; bonds in parallel beta-sheets are coloured dull yellow and can be recognized as a zig-zag pattern of bonds between parallel parts of the chain; the arrows drawn show the direction the chain takes, and can be displayed interactively upon request by the user (see section 4 · 5 · 3); the different sorts of loops can also be identified by the colours of the bonds. In the key displayed at the top right hand corner (choosing the LEGEND entry from the Polypeptide_Chain menu), the integer R (as previously defined) represents the distance as: the CO group of residue i binds to the NH group of residue i + R.

In Figure 4 · 10, the inter-mainchain hydrogen bonds involved in the actinidin protein (coded as 2ACT and where its X-ray crystallography was carried out by Kamphuis et al. [KAMPH 84]) are pictured. The lower domain, which is alpha-helical, is distinguished by the red bonds, and the top domain, being mainly beta-sheet, has yellow bonds.

4 · 5 · 2 · 2 Sidechain – Mainchain Hydrogen Bonds

For the display of sidechain — mainchain hydrogen bonds (SIDECHAIN HB option), each hydrogen bond is categorized according to its type, which is either a sidechain — mainchain bond or an inter-sidechain bond.

The polypeptide chain is represented as previously, and side chains involved in bonding are represented by tags which project from them. These tags are drawn with the same thickness and shade as those of the chain at that position. Their direction is computed by bisecting the obtuse angle formed by the smoothed alpha-carbon coordinates of the amino acid in question and those of its two neighbours. They are all the same length (1 Å) but may appear foreshortened or obscured.

Sidechain — mainchain interaction bonds are given two different colours depending on the interaction of the side chain with either the CO or NH group of the main chain. They are portrayed as thin lines between one tag and one 'smoothed' (averaged position using the averaging method) alpha carbon. Hydrogen bonds, where the sidechain bonds to a main chain group atoms (either NH or CO) of the same residue, are represented as thick oblique flag-like features using the same colour coding as for the other bonds.

For the inter-sidechain interactions, just one colour is allocated to the hydrogen bonds and they are portrayed as thin lines between the ends of the appropriate tags. Disulphide bonds are included as if they were inter-sidechain bonds. They are portrayed as thick lines between the ends of tags (cysteine side chains). They are also allocated a separate colour.

Figure 4 · 16 illustrates the sidechain — mainchain hydrogen bonds involved in the dihydrofolate protein (3DFR). Side chains involved in bonding are represented by white tags which project from the polypeptide chain. Hydrogen bonds are portrayed as lines, either between ends of tags (for inter-sidechain bonds, blue), or between one tag and one smoothed alpha-carbon (for sidechain — mainchain bonds, red or pink). On the right of the picture, one can see a thick pink oblique flag-like feature representing a hydrogen bond between a side chain and a main chain carbon atom of the same residue.

4 · 5 · 3 Direction of the Polypeptide Chain

In order to determine the direction of the polypeptide chain anywhere on the curve, the command ARROW has to be selected in the Polypeptide_Chain menu. Having chosen this option, the user can point to a specific region on the chain he wants an arrow to be drawn. This interactive command allows the user to detect the direction of the chain anywhere along its path. This is, for example, helpful for detecting N-termini in alpha-helices, or for observing if two strands of beta sheets are parallel or antiparallel. Figure 4 · 15 illustrates the effectiveness of such arrows for the dihydrofolate reductase protein. It is easy to see from the orientation of the (orange) arrows that strands of beta-sheets on the left of the image are antiparallel, while those on the right are parallel.

4 · 5 · 4 Piecewise Decomposition

Segments of the polypeptide chain such as domains can be selected

for display either by inputting their end segment amino acid number (echoed at the *Message Window)* or by pointing at them directly and clicking the mouse. Note that the amino acid numbers are the ones provided by the Brookhaven PBD. The command CHOOSE_SEGMENT from the Polypeptide_Chain menu performs this task. A maximum of four segments is allowed, due mainly to the diagrammatic views described later; in these later pictures, the display of more than four segments can clutter the display on the screen. If more are selected, only the first four are considered. These pictures are displayed with all appropriate hydrogen bonds involved in the segments (e.g. by removing any hydrogen bond that does not connect two amino acids belonging to one of the segments).

Figure 4 · 15 shows two fragments of the dihydrofolate reductase protein (3DFR) displayed with inter-mainchain hydrogen bonds. In the key displayed at the top right hand corner (choosing the LEGEND entry from the Polypeptide_Chain menu), the colour coding for all the hydrogen bonds involved is given. The LEGEND option also displays a line on the bottom left corner representing a one nanometer (= 10 Å) scale reference. In Figure 4 · 16, the two fragments are displayed with the sidechain hydrogen bonds involvement.

4 · 5 · 5 Manipulation

4 · 5 · 5 · 1 Rotation

The protein can be rotated through its center of mass in any one of the three axes by selecting the ROTATE command from the root menu. Then, automatically, the Rotate_submenu appears, to allow the user to select the direction of rotation. The last entry in this submenu, the

ANGLE option, allows rotation increments to be altered. These increments are 3, 5, 7, 10, 25, 45, 90 and 180º.

4 · 5 · 5 · 2 Scaling

Scaling the Polypeptide chain, either up or down, is possible by choosing the SCALE UP or SCALE DOWN entry, respectively, from the Polypeptide_Chain menu. A fixed scaling coefficient is applied to either enlarge or reduce the image by one third of its size. A more flexible tool to enter scaling factors would be to simulate a scaling ruler icon. As the user drags the cursor on a 'slider', a scaling number is displayed and scaling will be performed interactively.

4 · 5 · 6 Saving a View

After choosing the SAVE option from the root menu, the *Message Window* appears on the bottom right corner of the screen (unless previously positioned differently by the user), prompting the user to choose a new filename. This can be an alphanumeric string up to 10 characters long and can be of the form "3WGA". Some flexibility is allowed in the number of characters entered, but the user is recommended to employ the conventions of the Brookhaven PDB by using a four letter code. A long title (being the title displayed as a header in the *Polypeptide Chain Window)* to be stored with the data is then prompted for. This can be up to 80 characters long and can be of the form "wheat germ agglutinin". Control is then returned to the main menu. This is particularly helpful for a user wishing to save the last view of the polypeptide chain for another session.

4 · 5 · 7 Three-Dimensional Perception

In order to visualize the three-dimensional folding of the chain, an important question to solve is how to indicate depth in these polypeptide chains. Various methods are applied in these pictures in order to obtain good three-dimensional perception of the displayed model.

4 · 5 · 7 · 1 Depth Sorting

Depth sorting is applied to all line segments of the chain as well as the hydrogen bond lines. All lines are drawn in depth order according to their z-coordinates (with a small offset of 4 Å added for the hydrogen bond lines) with the farthest being drawn first. Also, the fact that thick circular brushes are used to draw these line segments (see section 4 · 5 · 7 · 4), implies that each line segment has a surface associated with it. Consequently, depth sorting on its own is not sufficient to guarantee the correct drawing order to achieve hidden surface removal. If 'depth overlaps' occur (see [NEWEL 72]), some additional comparisons are needed to determine whether any of the surfaces should be reordered. However, this test has not been performed in the present system, but should be considered to get a more accurate representation.

4 · 5 · 7 · 2 Intensity Depth Cueing

A range of eight shades of the same colour are used to represent intensity depth cueing, but only for the line segments (polypeptide chain and tags). This range is by no means a maximum limit, but it appears to show sufficiently the depth cueing effect on the chain. The line

segments near the viewer appear brighter than those far away. In Figure 4 · 6, the folding of 3TLN can be easily seen as the colour intensity of the chain varies from white to various shades of grey.

4 · 5 · 7 · 3 Tramlines

Sometimes a slight problem arises whenever line segments are very close to one another in depth. Depth cueing and depth sorting techniques alone are insufficient to indicate depth, especially in a still picture, since these line segments have the same colour intensity. This weakness is corrected by adding another level of depth perception, which we called the 'tramline' technique, closely related to the haloing technique [FOLEY 82]. It is applied to all line segments by surrounding the edges of their surface with a thin line, drawn in the background colour. Consequently, the relative depth of two line segments is indicated by a bridge effect when they cross. Note that this procedure is somewhat different from the hidden line removal technique [NEWMA 79] which concentrates exclusively on objects defined as a set of lines representing the edges of the surfaces of the object; while in our case, objects are defined just as a thick and unique line.

As the tramlines might take two to three seconds to be drawn, depending on the size of the protein under study, the user has the option (by clicking on the TRAMLINE or TRAMLINE OUT menu entry) to see or not the tramlines displayed. Commonly, tramlines are drawn in still pictures, allowing the reader to get a good depth perception of the model. If the model is being manipulated, then almost real time rotation through the center of mass can be obtained without tramlines. It is often easier to manipulate the model quickly without tramlines, until the

desired orientation of the chain is found. Then, tramlines can be restored in a still picture. Such pictures are best for publication purposes.

4 · 5 · 7 · 4 Circular Brushes

Finally, various brush shapes for drawing are implemented to indicate the relative depth of two consecutive line segments. Circular brushes are used to draw the line segments in all pictures. This gives rise to the appearance of a line segment being in front or behind its neighbouring one. The example in Figure 4 · 4 demonstrates this technique; in Figure 4 · 4 · a, line segment B is in front of line segment A, while in Figure 4 · 4 · b, it is the other way around. This method of display is effective, especially in displaying images of atomic details (see section 4 · 6).

Note that in the initial stages of the project, circular brushes have been implemented to ease tramline representation. But, as suggested in the last paragraph, they happen to be, by themselves, an enhancement to three-dimensional perception.

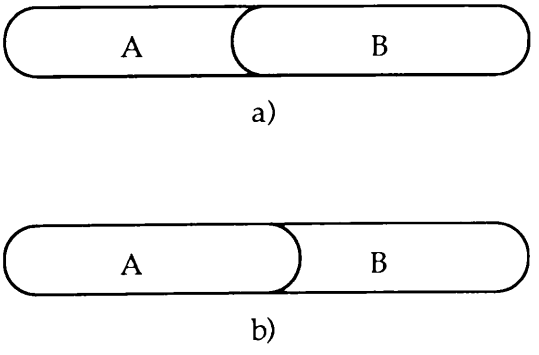
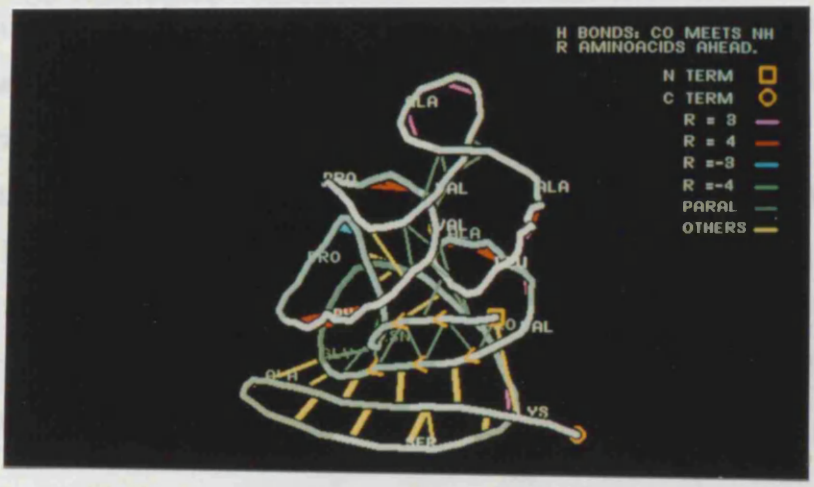


FIGURE 4 · 4 Relative Depth of 2 Consecutive Segments A and B

4 · 5 · 8 Background Colour

Whenever the polypeptide chain is manipulated, or any alteration in the image is requested by the user, the content of the window is first cleared out with a background colour before the image is redisplayed. This colour creates the background of the scene. It is set to black initially, but can be redefined by choosing successively, the BACKGROUND_COL entry from the Polypeptide_Chain menu and then the desired new background colour from a submenu having eight different colour entries. This command should be used with care, as a visual phenomenon called 'simultaneous colour contrast and colour mixture by averaging', may occur [AGOST 79]. If, for example, a green hydrogen bond line is drawn in two different backgrounds, one dull red and the other neutral gray, the line will not be perceived to be the same shade of green. Furthermore, an interesting remark by Orphir and al. [OPHIR 69] suggests that dark objects on a bright background provide a more restful and less fatiguing display. However, in the CG1 workstation made available to the research, video noise appears on the screen in the form of faint grey vertical lines, spaced every 8 pixels or so. There are visible on a light colour, but not black, background (see [ARTWI 84] for more information about register load noise). Figure 4 · 5 · a shows such noise effect on the screen as the polypeptide chain picture of 3DFR is displayed on a light green background. In Figure 4 · 5 · b, the same polypeptide chain is drawn with a black background. Thus, as a general rule, all the illustrative pictures in this thesis have a black background.

a/



b/

FIGURE 4 · 5 Comparison of Two Different Background Colours in the Display of the Three-Dimensional Structure of Dihydrofolate Reductase (3DFR)

Video noise appears on the screen in the form of faint grey vertical lines, spaced every 8 pixels or so. There are visible on the light green background in Figure 4 · 5 · a, but not on the black background of Figure 4 · 5 · b, where both pictures display the same dihydrofolate reductase protein. Notice, however, that the course of the chain is easier to follow with the light green background; this light colour provides a more restful and less fatiguing display (a remark made by Orphir and al. [OPHIR 69]).

4·5·9 Labelling

On choosing the LABEL option from the Polypeptide_Chain menu, labels of residues with either a three letter identifier (see appendix 1 for the identifier code) or the residue number given to them in the PDB protein file, can be displayed. Consequently, the label submenu appears on the screen containing two entries: LETTER and NUMBER. On choosing one of these two entries, a second-level submenu is displayed with predefined interval entries to choose from. These are: ALL, 5, 10, 20, 40, 50 and OTHERS. The ALL option displays all the amino acid labels. All the labels are depth-sorted and depth-cued along with the line segments forming the chain. They are drawn a little further out along the x axis from amino acid centers to ensure that the labels do not overwrite them. The intensity depth cueing of these labels uses the same intensity colour as for the line segments. In Figure 4 . 15 and Figure 4 . 16, labels are drawn along the main chain of the 3DFR protein. They are displayed as residue numbers and three-letter codes, respectively; spaced every 10 amino acids, these labels are depth sorted and depth cued.

4·6 ATOMIC DETAIL WINDOW

To study details, a close-up look facility is introduced in the system. This facility, seen as a zooming in of the polypeptide chain images, reveals all the constituting atoms of a particular segment. In this way, not only can turns be visualized, but every portion of the chain in its very detail (exact position of all the main and side chain atoms and all hydrogen bonds involved). The *Atomic Detail Window* is invoked for this purpose.

Furthermore, as noticed in the polypeptide chain pictures using the averaging method, some errors occur in turns, because the curve appears on the inside of the bend. The *Atomic Detail Window* is especially useful for the closer study of such regions, without loss of precision in the three-dimensional position of atoms. These turning regions are believed to provide a decisive influence in directing the folding of proteins ([LEWIS 71] & [ROSE 76]).

Using the *Atomic Detail Window*, any portion of the chain can be chosen. It can be specified by either pointing/clicking with the cursor and the mouse at the two end-points, or by typing the two numbers representing the alpha carbon numbers of the two segments end-points. The *Message Window* is automatically opened in the second case.

The *Atomic Detail Window* appears on the screen showing the atomic details of the chosen segment with the corresponding hydrogen bonds involved. It is initially rescaled to fit the window display. This child window is positioned at the bottom left corner of the screen (unless repositioned by the user), so that it does not obscure the polypeptide chain picture. This provides an effective way to investigate, simultaneously, the same region in two different contexts. The background window, here the *Polypeptide Chain Window*, because of the absence of atomic detail, provides an overall and a better presentation of the essential structure. In Figure 4 · 3, a general view of the display screen is shown where the *Atomic Detail Window* displays the atomic detail of a particular turn (from amino acid 35 to 38) of the thermolysin protein.

In these representations, line segments are also used to draw atoms, rather than, as in more conventional pictures, using ball and stick or sphere modeling. This enables some vision through the model. Moreover, these line segments are depth sorted to give a depth effect to

the model as this feature is completely lacking in simple stick model representation.

The line segments are drawn according to the following rule: all connecting bonds are drawn as line segments that run from the atom in question to a point half way between the two atoms being joined. Thus each bond consists of two segments, one contributed by each atom being joined. The use of various circular brush sizes to draw these line segments helps enhance the three-dimensional perception of the image, as seen already in Figure 4 · 4. The thickness of every drawing line is set as:

- very thin lines for hydrogen bonds

- thick line segments for connecting atoms in the backbone chain and less thick ones for the side chains.

Note that the connection between a side chain atom and a main chain (generally an alpha-carbon, and occasionally a nitrogen atom in case of a proline residue) is performed according to the same rule. That is, a thick line segment is drawn half way from the main chain atom to the side chain and a less thick one from the side to the main atom.

No special option is available to draw tramlines as they are always present. These tramlines do not take too much time to be drawn and, consequently, almost real time manipulation is performed on these pictures. However, special care has to be taken to draw tramlines in the case of side chain — main chain atom connection (see section 5 · 4 · 2 · 3

for more details on how to draw tramlines in such situation).

Hydrogen bond lines are depth sorted with respect to the z-coordinates of the line mid-points. No offset is added to these lines. All the other line segments (i.e. atoms) are depth sorted and depth cued by intensity. The process is determined in almost the same way as the one described earlier for the polypeptide chain. The only difference is the way depth cueing is performed on these pictures. Four different colours, with a range of three shades each, are used to distinguish between, respectively, carbon atoms on the main chain, carbon atoms on the side chains, oxygen atoms and nitrogen atoms. The reason for having separate colours for the carbon atoms, depending on whether they belong to the polypeptide chain or not, is that they are very common in proteins, as compared to nitrogen and oxygen atoms. I felt that, to avoid excessive use of colour while not losing information, only carbon atoms should be differently coloured (in addition to their thickness clue). This variation in thickness, together with the colouring of all the atoms, allows the user to differentiate, at a glance, which atoms belong to the polypeptide chain.

The user can either view all atoms constituting the portion of the chain (SIDE ATOMS entry), or the ones forming the backbone of that portion of the chain (MAIN ATOMS entry). The choice of these two options is illustrated in Figure 4 · 8 and Figure 4 · 9, where the atomic details of a portion of an alpha helix taken from the 3TLN protein are shown. In Figure 4 · 8, all the atoms are displayed together with the hydrogen bonds involved, while in Figure 4 · 9, the side chain atoms have been removed and only the backbone atoms are displayed. These two pictures also show the labelling of residues, either as a three-letter code, or as an amino acid number. The labelling commands are handled the same way as in the *Polypeptide Chain Window*. The only difference

is that, in these pictures, all the residues are labelled because of the available screen space, and the labels are positioned with a small shift on the right from the alpha-carbon atoms.

Choosing EXIT from the Atomic_Detail menu returns the user to the Polypeptide_Chain menu.

4·7 DIAGRAMMATIC WINDOW

A major problem in predicting three-dimensional structure from sequence lies in discerning the relationship between secondary structure and sequence. The diagrammatic pictures presented in this section give a visual aid for this purpose. In these computer-generated diagrams, the main chain is represented by horizontal row boxes and the hydrogen bonds by coloured lines. For consistency, the same colouring scheme as in the polypeptide chain pictures is used. As well as representing the polypeptide chain, the series of boxes show the sequence represented as single letters within these boxes (using the standard one-letter code given in appendix 1). The advantage of this type of display is that every hydrogen bond is clearly seen, since it cannot be obscured, as happens occasionally in the polypeptide chain pictures. Also, the relationship between bond pattern and sequence is evident.

4·7·1 Inter-Mainchain Hydrogen Bonds

When displaying inter-mainchain hydrogen bonds on these diagrams (MAINCHAIN HB option), hydrogen bonds are represented as pipe-like features joining appropriate boxes. The letter for each amino

acid (this option is described in section 4 · 7 · 6) is surrounded by a small box. To obtain the maximum display information without excessive visual detail, the boxes are conceptually divided into two parts. The left-hand side of the box corresponds to the NH part and the right-hand side to the CO part.

All hydrogen bonds are drawn with a thin line at one side or the other of the box to represent either NH or CO groups involved, except for paired bonds. Where there is a pair of hydrogen bonds between both sets of CO and NH, a thick line is drawn from the middle of each appropriate box (another consistency with the parent window display). Such thick pipe-like lines represent hydrogen bonds involved in antiparallel beta-sheets.

Another information display detail is added to hydrogen bonds involved in parallel beta sheets. In addition to the appropriate colour code allocated to them, a rectangular box is drawn at the center of the bonds forming a pair of strand within a parallel beta-sheet so that this feature can be readily identified. The rectangle length and height are variable and depend on, respectively, the length separating a pair of strands and the number of bonds involved in it. The more numerous the bonds are, the fatter the rectangle. The larger the number of amino acids separating the two strands, the longer the rectangle.

Depth sorting is applied to all the hydrogen bond lines. They are drawn in depth order according to their length (distance apart in sequence of two amino acids joined by a bond) with the longest being drawn first. Note that if depth sorting is performed the other way around, short pipe-like features can be overlapped in the vertical direction by fatter ones which would make them invisible. Horizontal overlapping is avoided by applying an algorithm described in section 5 · 4 · 5 · 1. Tramlines are

automatically drawn for every single object.

Figure 4 · 7 illustrates the diagrammatic views of inter-mainchain hydrogen bonds in 3TLN. The horizontal rows of boxes (315 in total) represent the amino acid residues. Inter-mainchain hydrogen bonds are represented as coloured pipe-like features between appropriate residues. Note that a rectangular box is drawn at the center of the bonds forming a pair of strands within a parallel beta-sheet so that such features can be readily identified; the three rectangles indicate that this protein has 3 parallel beta-sheets (not seen distinctively in the polypeptide chain pictures in Figure 4 · 6). As in the other pictures, red bonds are mainly alpha-helices (7 in total), the yellow ones beta-sheet, and the other colours mostly represent different sorts of loop motifs.

4 · 7 · 2 Side Chain – Mainchain Hydrogen Bonds

For the diagrams involving sidechain – mainchain hydrogen bonds, slight transformations are made in the way hydrogen bonds are displayed. While the main chain is still represented as horizontal boxes, with letters in boxes representing amino acids as in the previous diagrams, side chains involved in bonding are represented by vertical lines (tags) drawn from the middle of each of the appropriate boxes. The length of each tag approximates the length of the corresponding side chain; lengths of tags are given in Appendix 2.

Hydrogen bonds are portrayed as coloured lines, either between the ends of tags (for inter-sidechain bonds), or between one tag and one main chain 'box' (for sidechain — mainchain bonds). For sidechain — mainchain hydrogen bonds, the main chain connector is drawn from the

middle of the box since the two colour codes allocated to the NH and CO groups are sufficient to distinguish between these two groups.

Hydrogen bonds between side chain and main chain atoms from the same residue are portrayed as thick oblique flag-like features (note the consistency with the parent window displays). In addition to the two colour codes allocated to this type of hydrogen bond, a left-handed flag indicates an NH group involvement, while a right-handed flag indicates a CO group.

Apart from the flag-like line drawings, all the other hydrogen bonds are drawn either as straight or bent lines joining the appropriate residues. In the case of inter-sidechain bonding involving two residues of the same type but distant in sequence, a straight line is drawn (since they have the same tag length). In all the remaining cases, bent lines are drawn. When a side chain is hydrogen bonded to more than one other atom, the lines departing from the tag are drawn in a 'stacked' manner to avoid superposition of line drawing. Tramlines are drawn automatically, and depth sorting is applied according to the same rules as the inter-mainchain bonds.

In Figure 4 · 18, the diagrammatic view of the same fragments as in Figure 4 · 17 is displayed, but with sidechain hydrogen bonds involvement. Side chains involved in bonding are represented by white vertical lines, their length approximating to the length of the side chain. Hydrogen bonds are portrayed as coloured lines, either between the ends of tags (for inter-sidechain bonds, blue), or between one tag and one main chain 'box' (for sidechain — mainchain bonds, red or pink). Hydrogen bonds between side chain and main chain atoms from the same residue are portrayed as thick oblique flag-like features (one is seen as a pink thick right-handed flag at amino acid 48 in the upper region). Note the

hydrogen bond connections between side chain 125 and 126 in the lower fragment; the blue lines departing from either tag are drawn in a 'stacked' manner to avoid superposition of line drawing.

4 · 7 · 3 Simultaneous Display of Two Protein Sequences

A user can compare diagrammatic views of the protein under investigation with any other protein. On choosing the "NEW_PROTEIN" entry from the parent menu, the *Message Window* appears and prompts the user to enter the new protein code. Initially, the *Diagrammatic Window* screen is divided into, essentially, two halves. The upper half is occupied by the diagrammatic view of the first protein (one horizontal row of boxes representing the whole protein), the lower half by the second. The possibility of inverting the orientation of the hydrogen bonds in the second diagrammatic view was considered. It would facilitate direct comparison of sequences versus hydrogen bonding for two such rows, but was discarded as being too cumbersome if three or four segments are compared simultaneously (e.g. for comparison of several domains). The way to display several segments for comparison is explained in the next section (section 4 · 7 · 4). Figure 4 · 11 shows the simultaneous display of two closely similar proteins with their inter-mainchain hydrogen bonds. Actinidin (coded as 2ACT and where its X-ray crystallography was performed by Baker & Dodson [BAKER 80]) is at the top, and papain (coded as 9PAP and where its X-ray crystallography was carried out by Kamphuis et al. [KAMPH 84]) is in the lower diagram. They exhibit 50% identity at the sequence level, but are more related at the level of inter-mainchain hydrogen bonds. Note the red alpha helical bonds and the yellow beta sheet ones. Note also that the horizontal rows of boxes are coloured differently; the upper one is white

and the lower one is dull yellow.

4 · 7 · 4 Piecewise Decomposition

As in the polypeptide chain pictures, user specification of the display of parts of the chain (or chains for the simultaneous display of two proteins) is performed either by entering amino acid number of the end segments (prompted/echoed at the *Message Window*) or by pointing/clicking at the appropriate boxes. The corresponding segments are displayed as a series of rows, each representing a segment. A maximum of four segments is allowed. With this facility, it is possible, for example, to align particular segments so that important similarities in sequence, as well as bonding patterns, can be focused upon. Figure 4 · 12, Figure 4 · 13 and Figure 4 · 14 illustrate this fact by displaying two segments for comparison, taken from the simultaneous display of the two proteins in Figure 4 · 11. In Figure 4 · 12, the upper segment represents the actinidin chain from amino acid 47 to 89 (white horizontal row of boxes), while the lower segment represents the papain chain from the same amino acid sequence (dull yellow horizontal row of boxes). Figure 4 · 13 displays the same regions as in Figure 4 · 12, but with the sidechain hydrogen bonds involvement. From these two pictures, it is evident that the two regions exhibit very high identity at the sequence level and are more related at the level of inter-mainchain hydrogen bonds. In Figure 4 · 14, the two segments are not taken from the same amino acid sequence (the actinidin region is taken from amino acid 151 to 198, whereas, the papain region is from 144 to 191) but are still very related at the level of inter-mainchain hydrogen bonding.

Figure 4 · 17 and Figure 4 · 18 show the diagrammatic views of, respectively, the inter-mainchain and the sidechain —

mainchain hydrogen bonds of two fragments taken from a single protein (3DFR).

4·7·5 Manipulation

4·7·5·1 Translation

The polypeptide chain(s) represented as one (or several) horizontal(s) row(s) of boxes can be translated horizontally, either to the right or the left in order to align particular segments. The manipulation is performed by choosing, respectively, from the root menu the "TRANSLATE —>" option or the "TRANSLATE <—" option. The choice of the translation factor is a multiple of the size of an amino acid box, and is under user control. On choosing the TRANSL_FACT from the root menu, a submenu will appear in order for the user to specify this factor. The different entries are : 1, 5, 10, 15 and 20. The factor is set initially to 5.

4·7·5·2 Scaling

As in the polypeptide chain pictures, the diagrammatic views can be scaled either up or down by selecting, respectively, the SCALE UP or SCALE DOWN option from the parent menu. Here also, only one scaling factor is allowed, which enlarge or reduce the picture by one third of its size. This option is used frequently in order to zoom in on a simultaneous display of several segments and study homology in greater detail.

4 · 7 · 6 Labelling

The LABEL command, from the root menu, labels residues with either the standard one-letter code identifier (see appendix 1) or the residue number. The letter code labelling option displays a character (code identifier) in every amino acid box. This facility enables a user to compare, for example, the relationship between bond pattern and sequence easily. The residue number labelling option displays amino acid numbers above the main chain boxes as markers spaced, every x amino acid boxes; x being a predefined width interval chosen from 1, 5, 10, 20, 40 or 50. The choice of the intervals is under user control.

4 · 8 MESSAGE WINDOW

The Message Window is the area for displaying error messages, prompts and feedback messages. Initially, this window is positioned at the bottom right corner of the screen, as seen in Figure 4 · 3, but can be manipulated freely by the user, as the other windows. This window comes (unstow the window) and goes (stow the window) automatically as needed. Input from the user is echoed in this window so that errors can be detected; backtracking is available to correct these errors.

4 · 9 COLOUR KEY WINDOW

In this window, a legend with the colour key for every hydrogen bond involved in, either the parent or the child windows, is displayed. It displays either the inter-mainchain or the sidechain — mainchain hydrogen bond colour code, depending on the type of picture involved.

Upon choosing the COLOUR_KEY entry, from any root menu, this window appears, usually at the bottom right corner of the screen.

4 · 9 · 1 Inter-Mainchain Hydrogen Bonds

For the inter-mainchain hydrogen bonds, the colour key gives the colour scheme used to distinguish different types of hydrogen bonds according to the relative sequence positions of the amino acids, joined by NH and CO groups. The rule to determine the value of R (the difference between the amino acid sequence numbers of the NH and CO groups involved) is as follows: the CO group of residue i binds to the NH group of residue i + R.

A variety of colours is used for the bond types, but in a conservative manner. For instance, when R is between - 2 and + 2, only one colour is allocated. Furthermore, bonds between amino acids not near in sequence (N > 5 or N < - 5) are given one colour. An exception in this category is made for bonds that belong to parallel beta sheets as they are allocated a separate colour; this is to emphasize their presence and distinguish them from the other bonds. Other bonds in this group, the hydrogen bonds involved in antiparallel beta sheets, do not need an individual colour. They are sufficiently distinct from the others, as a thick line portrays them, both in the polypeptide chain pictures and the diagrammatic views.

Figure 4 · 3 shows an instance of such colour scheme for the 3TLN protein displayed as a three-dimensional structure. Note that in the legend, the value R = - 5 is missing. This is due to the fact that no bonding exists between any CO group of residue i and NH group of residue i - 5. This may help, for instance, a user to detect, in a glance,

which types of bonding occur in a given protein.

4 · 9 · 2 Sidechain – Mainchain Hydrogen Bonds

For the inter-sidechain interactions, just one colour is allocated to the hydrogen bonds and is labelled as "SIDE-SIDE" in the *Colour Key Window* legend. Disulphide bonds ("DISULPHIDE" label), if they exist, have a separate colour. Sidechain — mainchain interaction bonds are given two different colours depending on the interaction of the side chain with either the CO or NH group of the main chain.

4 · 9 · 3 N - Terminus and C - Terminus

The N-terminal and C-terminal amino acids are represented, respectively, by a square and a circle symbol of the same colour. The same colour is used to represent arrows in the polypeptide chain pictures (see section 4 · 5 · 3) to retain consistency, as they all indicate direction. In Figure 4 · 3, the N-terminus and C-terminus are represented in the *Colour Key Window* as an orange square and circle, respectively.

4 · 9 · 4 Colour Editor

Colour coding of hydrogen bonds is under user control. He may change one particular colour scheme until he is satisfied with his design. The way a user changes the colour of a type of bond is done the following way: whenever the user points/clicks at a particular colour box, the box displays a new colour. The same process can be performed for all the

other bonds. Finally, the DONE box is selected to acknowledge satisfaction with a particular colour scheme.

4·10 HOW TO START A SESSION ?

To start a session, a user has first to log in to the system. Then, once the shell script appears on the screen (usually a $ sign), acknowledging valid access to the system, two parameters have to be entered: the program name "PHD" which initiates execution of the main protein visualization program, followed by the code corresponding to the protein to be investigated (usually a four letter code). For example, a possible entry would be : "PHD 3WGA". Some preprocessing takes place (reading atomic and hydrogen bond files, structuring internal operations, etc ...) before the *Polypeptide Chain Window* is displayed showing the polypeptide chain picture with the inter-mainchain hydrogen bonds. The chain is as large as possible, yet still fits into the display area of the window. The averaging factor is set initially to 5. The Polypeptide_Chain menu then appears at the top right-hand corner (unless specified otherwise by the user, initially), and the user is in a position to start communicating with the system.

4·11 CONCLUSION

It is hoped that the description of the operations at the user level and the illustrative pictures taken directly from the screen show that the aims of the PHD software have been fulfilled. This software is useful in assisting a scientist to understand more about the three-dimensional structure of proteins and the relationships between them and their

sequence. It is easily used, even by a beginner. Perceptual response of the eye to visual information is performed efficiently. Effective colour display highlights important features found in proteins. Depth perception in these images have made it possible to create models of proteins which stimulate the imagination of the end-user, while minimizing time delays for display or manipulation of the image.

In the polypeptide chain computer-generated pictures, as well as in those for atomic detail, all the three-dimensional hydrogen bond patterns of loops and secondary structural features are displayed. On the other hand, the diagrammatic views display hydrogen bonds in relation to the primary structure, rather than in the context of the three-dimensional structure. Several advantages emerge from the diagrammatic views. In this type of display, every hydrogen bond is clearly seen, since it cannot be obscured by other bonds or the path of the chain, as happens occasionally in the polypeptide chain pictures. Yet, a major advantage is that the relationship between bond pattern (secondary structures, turns, loop motifs) and sequence is evident. This is likely to be especially useful for studies aimed at predicting the three-dimensional structure of proteins from sequence information.

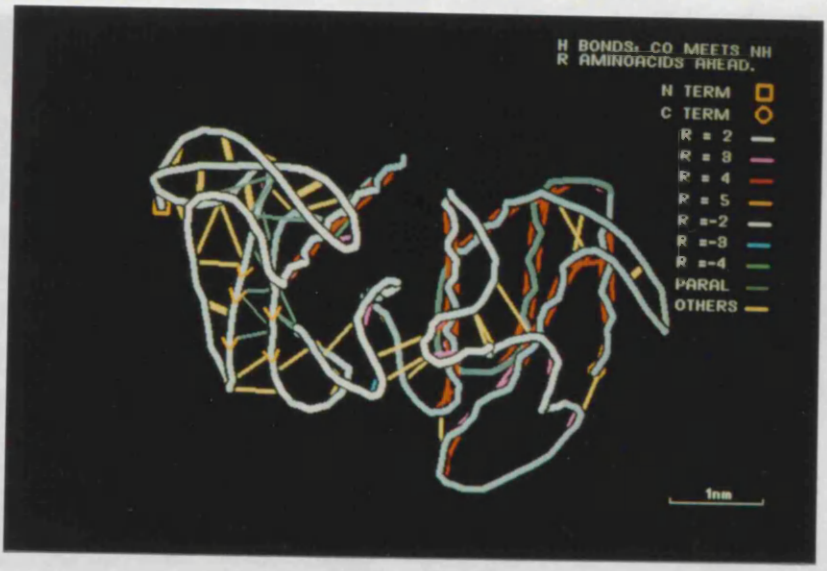


FIGURE 4 · 6 Inter-Mainchain Hydrogen Bonding in the Three-Dimensional Structure of Thermolysin (3TLN)

The thick white line represents a smoothed alpha-carbon plot. Inter-mainchain hydrogen bonds are drawn as coloured lines joining the positions of appropriate main chain atoms. Where both NH and CO groups of one amino acid are joined to the CO and NH groups of one other amino acid, a thicker coloured line is drawn. The colour represents the distance apart in the sequence of two amino acids joined by a bond; in the colour key shown at the top right-hand corner, an integer R defines this distance as: the CO group of residue i binds to the NH group of residue i + R. Alpha-helices appear as wavy lines, with red hydrogen bonds; hydrogen bonds in antiparallel beta-sheets are yellow; hydrogen bonds in parallel beta-sheets are coloured dark green; and the different sorts of loops can be identified by the colour of the bonds. The protein coordinates are from X-ray crystallography work by Holmes and Matthews [HOLME 82].

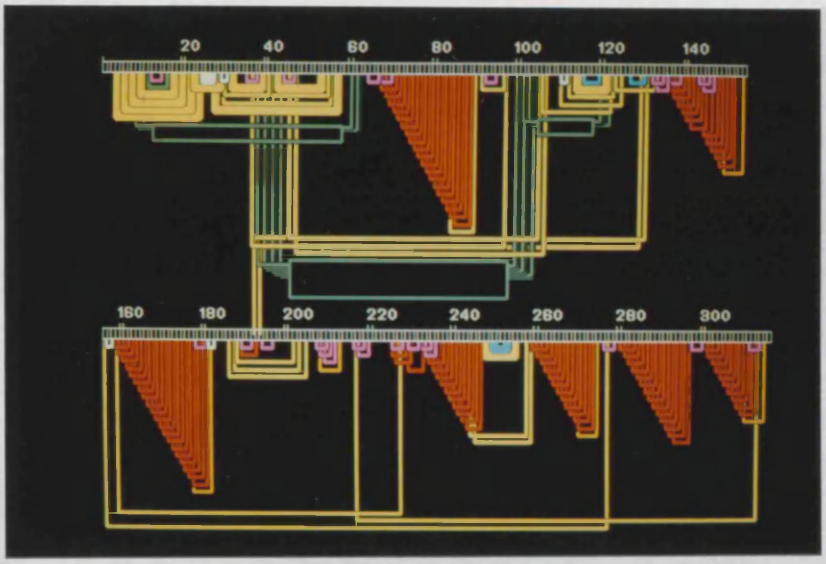


FIGURE 4 · 7 Computer-Generated Diagram of Inter-Mainchain Hydrogen Bonds in Thermolysin (3TLN)

The horizontal rows of boxes represent the amino acid residues. Inter-mainchain hydrogen bonds are represented as coloured pipe-like features between appropriate residues, which connect at the right or left of each box depending on whether the NH or CO group is involved. A rectangular box is drawn at the center of the bonds forming a pair of strands within a parallel beta-sheet so that such features can be readily identified. As in the other picture (the polypeptide chain figure), red bonds are mainly alpha-helices, the yellow ones antiparallel beta-sheets, and the other colours mostly represent different sorts of loop.

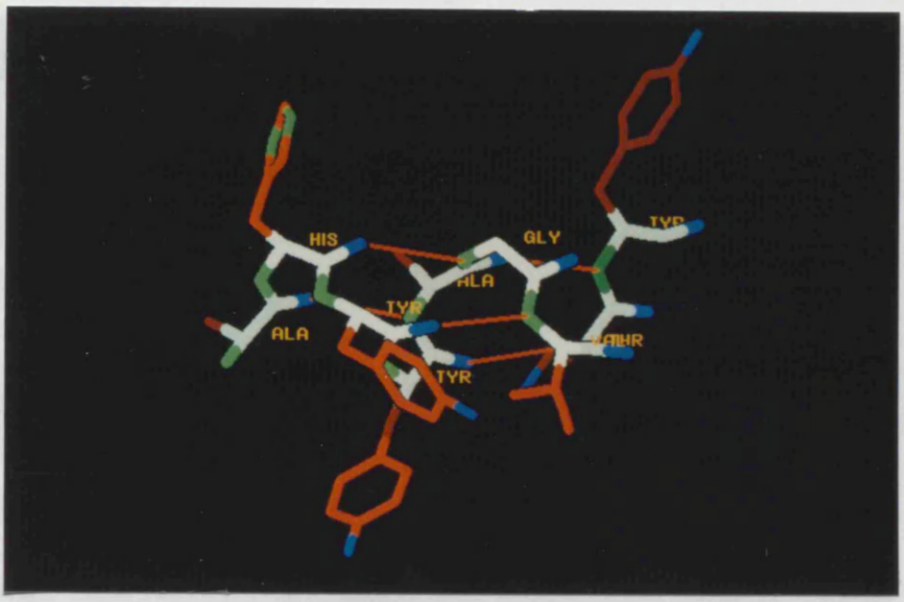


FIGURE 4 · 8 Atomic Details of a Portion of Thermolysin (3TLN) Displaying All the Atoms

A portion of an alpha helix of 3TLN (from amino acid 73-81) is shown. Note the red alpha helical hydrogen bonds forming the helix. The green-shade line segments represent nitrogen atoms; the blue-shade line segments are for oxygen; the white-shade ones are for carbon atoms on the main chain; and the red-shaded ones are for carbon atoms on the side chain. Labelling of residues as a three-letter code is displayed for every amino acid; the labels are depth sorted along the other line segments forming the picture.

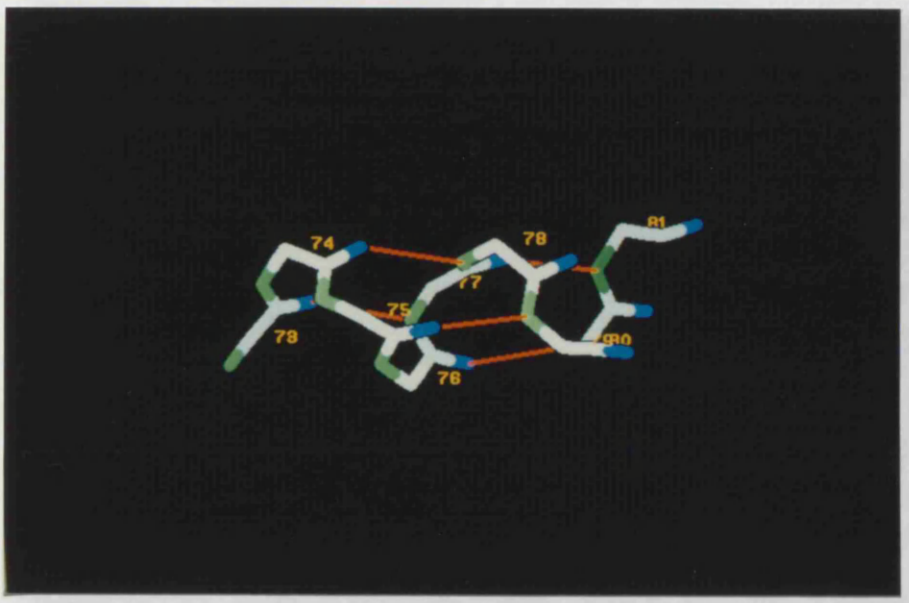


FIGURE 4 · 9 Atomic Details of a Portion of Thermolysin (3TLN) Displaying Only the Backbone Atoms

The side chain atoms from Figure 4 · 8 are removed and only the backbone atoms of the helix are shown together with the red hydrogen bonds joining the appropriate Oxygen atoms (in blue) to the Nitrogen atoms (in green). Labelling by amino acid number is shown for every residue.

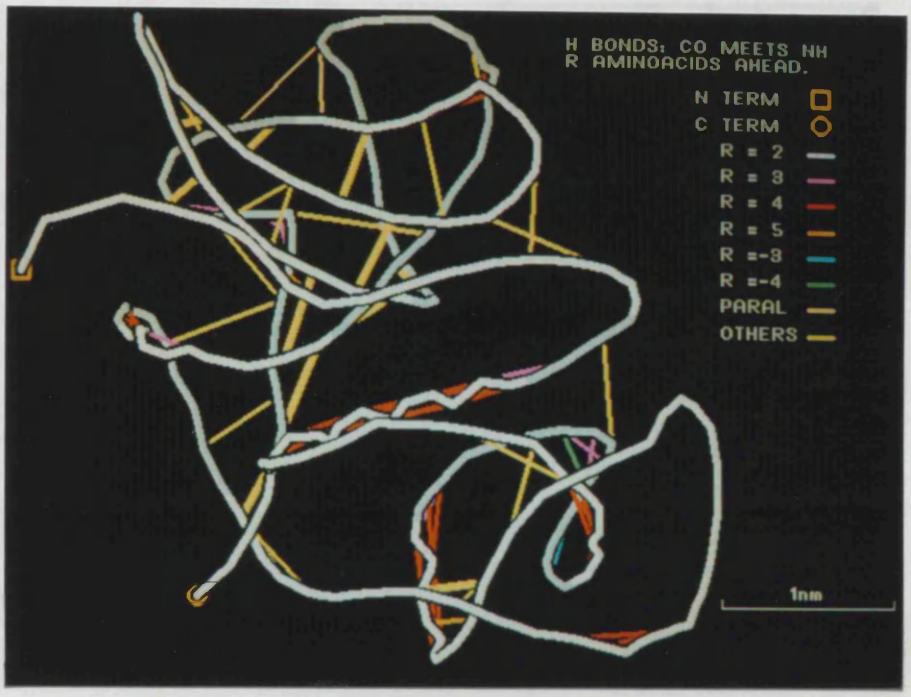


FIGURE 4 · 10 Inter-Mainchain Hydrogen Bonding in the Three-Dimensional Structure of Actinidin (2ACT)

The X-ray crystallography of 2ACT was carried out by Kamphuis et al. [KAMPH 84]. The thick white and grey line represent a smoothed alpha-carbon plot, and inter-mainchain hydrogen bonds are drawn as coloured lines joining appropriate smoothed alpha-carbon atoms. The lower domain, which is alpha-helical, is distinguished by the red bonds, and the top domain, being mainly beta-sheet, has yellow bonds.

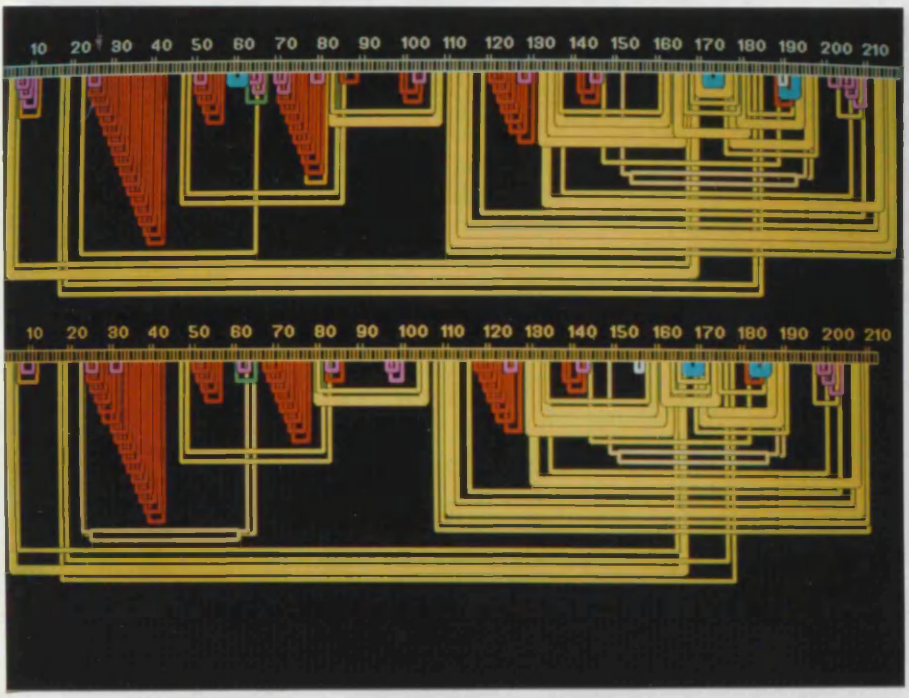


FIGURE 4 · 11 Diagram Comparing Inter-Mainchain Hydrogen Bonds in Actinidin (2ACT) and Papain (9PAP)

2ACT (X-ray crystallography by Baker & Dodson [BAKER 80]) is at the top, and 9PAP (X-ray crystallography by Kamphuis et al. [KAMPH 84]) is in the lower, diagram. They exhibit 50% identity at the sequence level, but are more related at the level of inter-mainchain hydrogen bonds. Note the red alpha helical bonds and the yellow beta sheet ones.

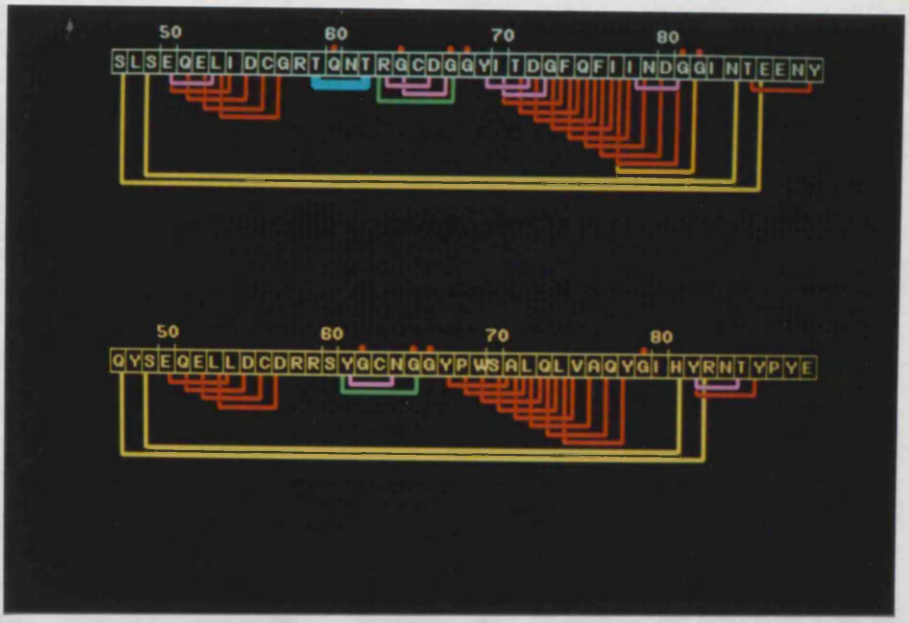


FIGURE 4 · 12 Diagram Comparing Inter-Mainchain Hydrogen Bonds in Homologous Regions of Actinidin (2ACT) and Papain (9PAP), Both Taken from the Same Amino Acid Sequence Covering Amino Acid 47 to 89

The lower diagram is 9PAP and the upper one 2ACT. The sequence is given as the standard one-letter code. Hydrogen bonds are drawn in colour in a pipe-like representation. As before, where NH and CO groups of a pair of amino acids are bonded, a single thick line is drawn. The letter for each amino acid is surrounded by a small box. The left-hand side of the box corresponds to the NH part and the right-hand side to the CO part. Hydrogen bonds are drawn at one side or the other of the box to represent NH or CO groups. A few atoms, for example the oxygen of GLU 50 in 2ACT, bond to more than one other atom. Whenever dihedral angles are greater than zero, a red dot is displayed above their corresponding 'boxes'. The two (red) alpha helices and the (purple and green) beta-bulge loops are clearly homologous, but the short turquoise beta-hairpin in 2ACT appears to have been inserted (or deleted in 9PAP).

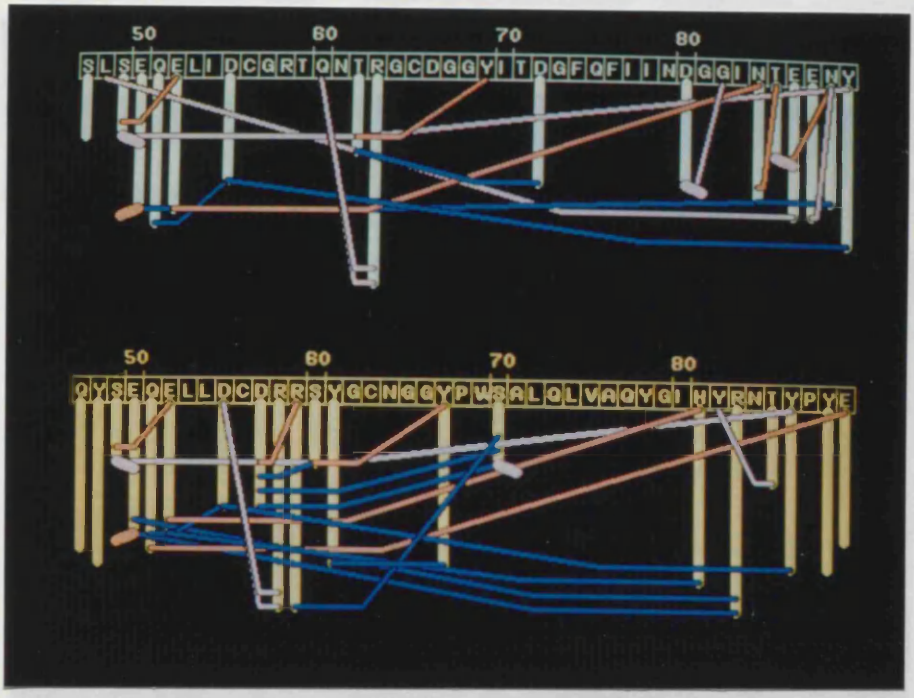


FIGURE 4 · 13 Diagram Comparing Sidechain Hydrogen Bonds in Homologous Regions of Actinidin (2ACT) and Papain (9PAP), Both Taken from the Same Amino Acid Sequence Covering Amino Acid 47 to 89

The higher diagram is 2ACT and the lower one 9PAP. Side chains involved in bonding are represented by either white vertical lines (in 2ACT) or dull yellow (in 9PAP), their length approximating to the length of the side chain. Hydrogen bonds are portrayed as coloured lines, either between the ends of tags (for inter-sidechain bonds, blue), or between one tag and one main chain 'box' (for sidechain — mainchain bonds, orange or lilac). Hydrogen bonds between side chain and main chain atoms from the same residue are portrayed as thick oblique flag-like features (one is seen at SER 70 in the 9PAP region).

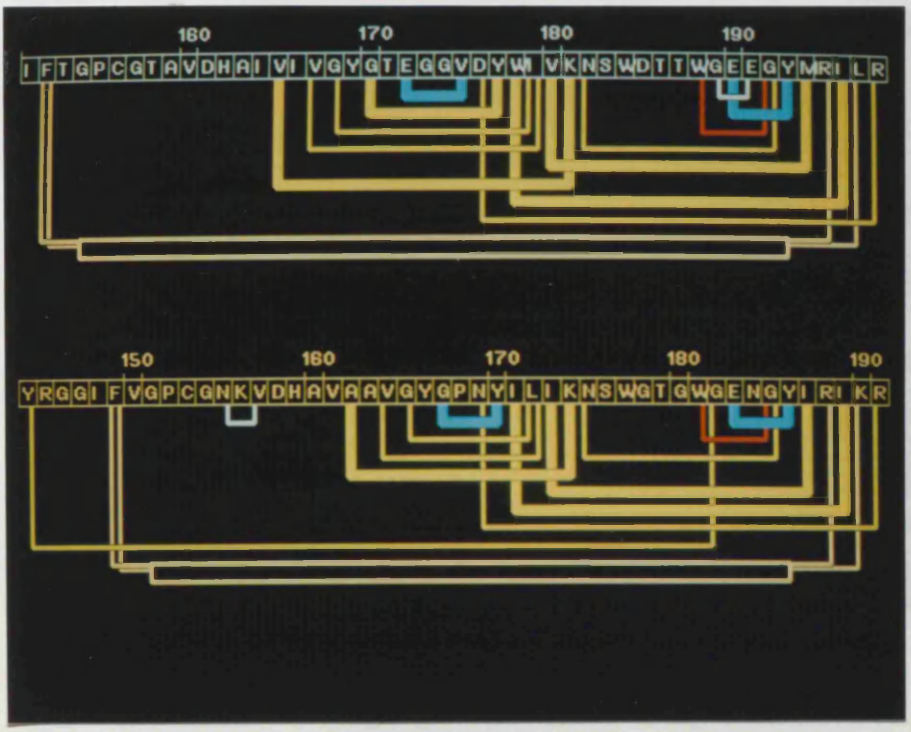


FIGURE 4 · 14 Diagram Comparing Inter-Mainchain Hydrogen Bonds in Homologous Regions of Actinidin (2ACT) and Papain (9PAP) Taken from Different Amino Acid Sequences

The lower diagram is 9PAP and the higher one 2ACT. The 2ACT region is taken from amino acid 151 to 198, while the 9PAP region is from 144 to 191. The turquoise bonds indicate the positions of beta turns at the loop ends of the two (yellow) beta-hairpins in both proteins. Note the hollow rectangular box in both regions, drawn at the center of the bonds, forming a pair of strands within a parallel beta sheet. Although homology between these proteins is high, an insertion or deletion, situated in the hairpin in the middle of the picture, is needed to fully align the proteins.

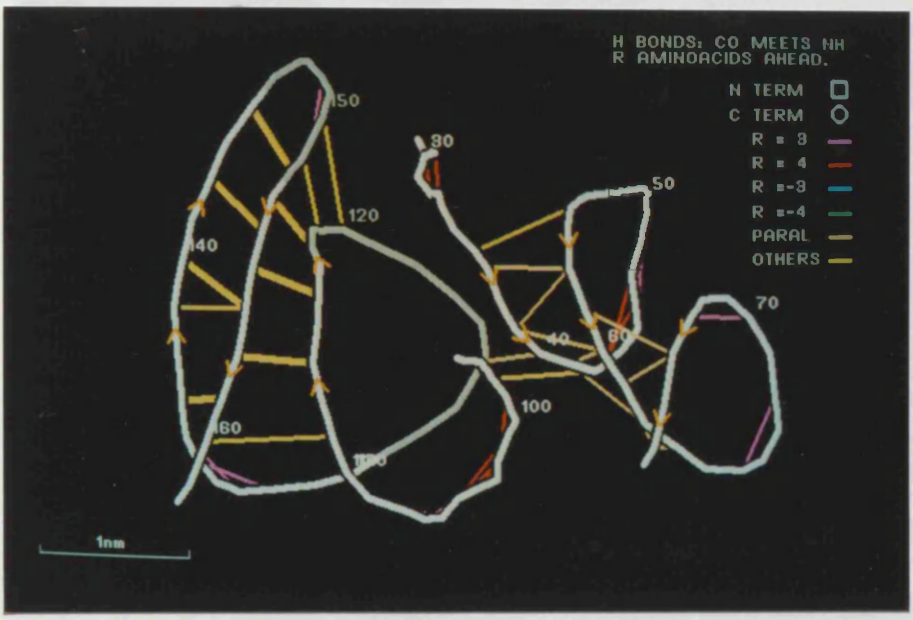


FIGURE 4 · 15 Piecewise Decomposition of the Polypeptide Chain of Dihydrofolate Reductase (3DFR) With Inter-Mainchain Hydrogen Bonds Display

The X-ray crystallography of 3DFR was carried out by Filman et al. [FILMA 82]. Two fragments of 3DFR, from amino acid 28-76 and from 97-161, are displayed simultaneously. Inter-mainchain hydrogen bonds involved only in these two regions are displayed. The colour coding for these hydrogen bonds is given in the key displayed at the top right hand corner. The orange arrows indicate the direction of the chain along its path. It is easy to see from the orientation of the arrows that strands of beta sheets on the left fragment are antiparallel while those on the right are parallel. Labelling is performed on residue numbers spaced every 10 amino acids. These labels are depth sorted and depth cued along with the line segments forming the polypeptide chain.

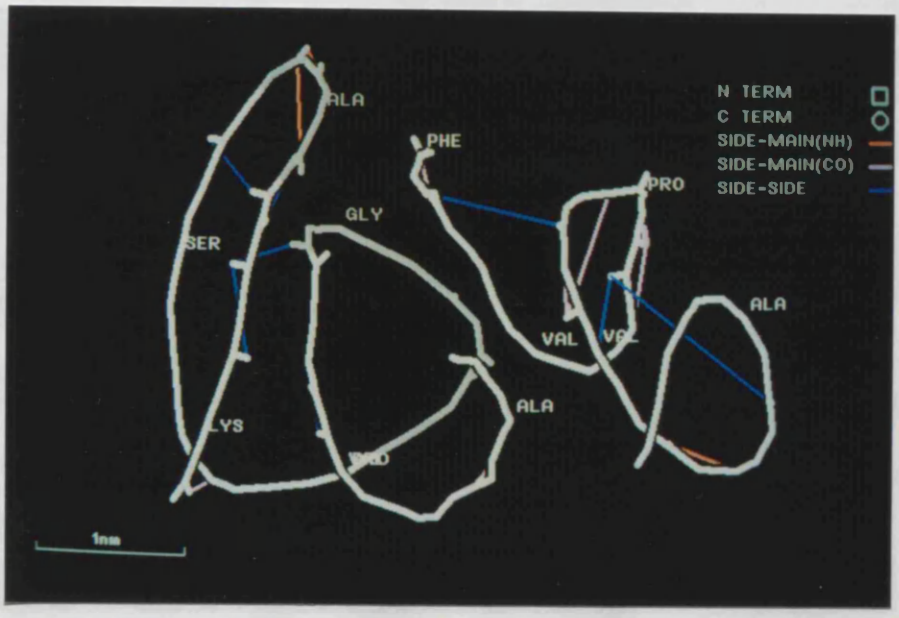


FIGURE 4 · 16 Piecewise Decomposition of the Polypeptide Chain of Dihydrofolate Reductase (3DFR) With Sidechain Hydrogen Bonds Display

The two polypeptide chain fragments of 3DFR are represented as in Figure 4 · 15. Sidechains involved in bonding are represented by tags which project from the polypeptide chain. Hydrogen bonds are portrayed as lines of other colours, either between the ends of tags (for inter-sidechain bonds, blue), or between one tag and one smoothed alpha-carbon (for sidechain — mainchain bonds, red or pink). Hydrogen bonds, where the side chain bonds to a main chain atom of the same residue, are represented as thick oblique flag-like features using the same colour-coding as for the other bonds. One can be seen on the right fragment as a pink flag. The three-letter code labelling is displayed; spaced every 10 amino acids, these labels are depth sorted and depth cued.

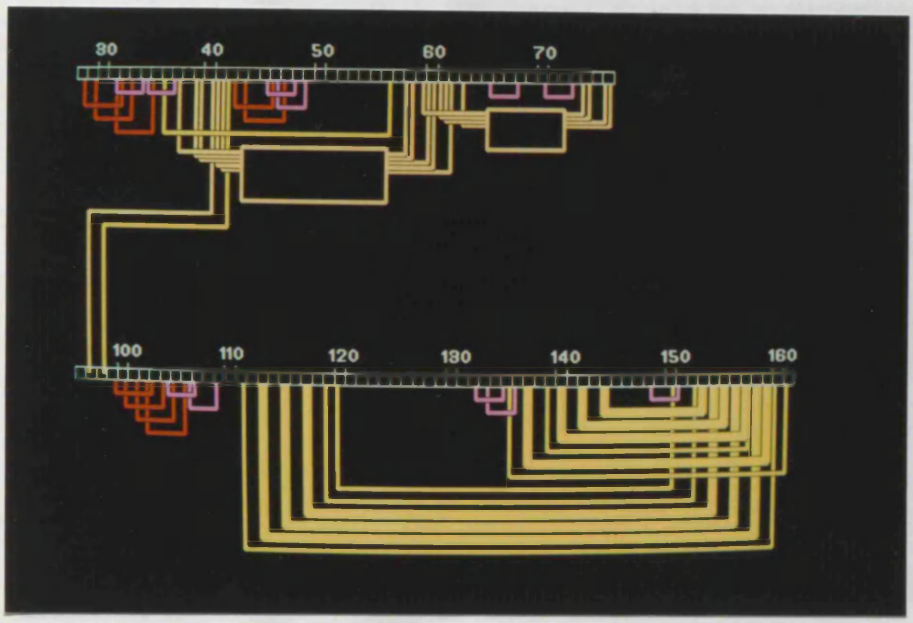


FIGURE 4 · 17 Diagram of the Inter-Mainchain Hydrogen Bonds of Two Fragments of Dihydrofolate Reductase (3DFR)

The higher diagram corresponds to the right fragment in Figure 4 · 15 (amino acid 28-76) while the lower diagram represents the left fragment(96-161). Hydrogen bonds are drawn in colour in a pipe-like representation. As before, where NH and CO groups of a pair of amino acids are bonded, a single thick line is drawn. Note the two hollow rectangular boxes in the upper region, drawn at the center of the bonds forming pairs of parallel strands; the lower regions is mainly composed of antiparallel beta sheets represented as thick yellow 'pipes'.

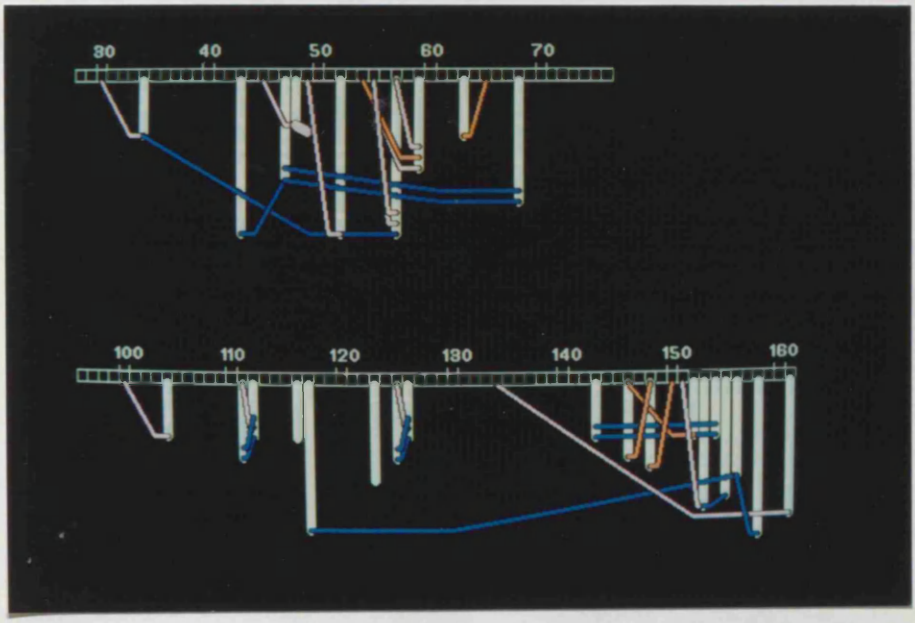


FIGURE 4 · 18 Diagram of the Hydrogen Bonds Involving Side Chains in Two Fragments of Dihydrofolate Reductase (3DFR)

The diagrammatic view of the same fragments as in Figure 4 · 17 is displayed but with sidechain hydrogen bonds involvement. Sidechains involved in bonding are represented by white vertical lines, their length approximating to the length of the side chain. Hydrogen bonds are portrayed as coloured lines, either between the ends of tags (for inter-sidechain bonds, blue), or between one tag and one main chain 'box' (for sidechain — mainchain bonds, red or pink). Hydrogen bonds between side chain and main chain atoms from the same residue are portrayed as thick oblique flag-like features (one is seen at amino acid 48 in the upper region).

Chapter 5

INTERNAL STRUCTURE

5·1 INTRODUCTION

In writing a molecular software package, which is similar to a relatively large software development project, part of the work has to be devoted to researching algorithms and designing the structure of the software. Once the design is determined and the algorithms, data structures, and modules are chosen, coding the program is relatively straightforward. While Chapter 4 was concerned with the design of the user interface to the system, this chapter is devoted to the design of the internal structure of the PHD software package as well as describing the various algorithms.

5·2 HARDWARE EQUIPMENT

The Whitechapel Colour Graphics Workstation CG1 is based on a National Semiconductor 32016. Some important characteristics of the machine are as follows:

- A raster display screen, with 768 x 576 resolution, is refreshed directly from main memory and thus does not need any intermediate frame buffer. The refresh rate is 57 Hz.

- A Video output which has only a 6-bit output (2 bits of control over each of the red, green and blue channels) and thus making the display device generate no more than 64 different colours from a palette of 256 possible colours.

- A hardware cursor, where a 65 x 65 bitmap from memory can be video mixed with the main screen image.

The CG1 runs on GENIX (a variant of the UNIX operating system) providing a multi-window UNIX environment, supporting both button mouse and keyboard input.

5·3 SOFTWARE DESIGN METHODOLOGY

In the design of the PHD system, there exist 3 distinct phases . This idea follows the design methodology used by Miller in his model

building system BUILD [MILLE 79] where he describes his three phases as follows:

> *<< The first deals strictly with internal requirements and is prompted by posing the question: "what are the basic operations which must be performed on the data?". Answering this question leads to the development of the data structures and algorithms to be used in the implementation of the system.>>*

> *<<...The second phase must be dealt with independently from the first and is prompted by asking the question: "what is the most natural way for the user to invoke these operations?. This determines what the user interface to the system should be. The order in which the two phases are addressed does not seem to be very important as long as decisions made in one phase do not adversely affect those made in the other.>>*

> *<<...The third development phase is ideally a relatively easy one: translating user action and/or commands defined in phase 2 into appropriate calls on algorithms developed in phase 1.>>*

The second phase was described in Chapter 4. The core of this chapter is devoted to phase 1, while in section 5 · 5, some points are given that need to be taken into account when dealing with phase 3.

5 · 4 DATA STRUCTURES AND ALGORITHMS

5 · 4 · 1 Data Structure for Input Files

5 · 4 · 1 · 1 Atomic Coordinate File

When a protein, say x, is chosen to be visualized, the first step is to store in core memory its atomic coordinates together with their attributes; these are read from the "x.mol" file, described in Chapter 3. A linear list, *chain[ ]*, is allocated for such storage. Each entry, ATOM, has the data structure shown in Figure 5 · 1. The first field is used for atomic coordinates (*coord*), followed by the atom identifier (*atom_type*), amino acid type (*amino_type*), true atom sequence number (*atom_num*) and true amino acid sequence number (*amino_num*). Note that the total number of atoms, declared as Tol_Num_Atom and already specified at the beginning of the "x.mol" file, makes dynamic memory allocation feasible for such storage. An appropriate scaling and a translation to the center of mass are then performed for this table, so that the whole image can be displayed on the screen.

coord			atom_type	amino_type	atom_num	amino_ num
X	Y	Z				
3.5	2.4	1.1	CA	ALA	23	8

FIGURE 5 · 1 Data Structure of ATOM Record

5 · 4 · 1 · 2 Hydrogen Bond File

As the corresponding hydrogen bond file "x.hbond" is read, all relevant hydrogen bond information is stored in a linear list, *hbond[ ]*; an example of an entry, HBOND, is shown in Figure 5 · 2. Tol_Num_Hbond, already defined in the "x.hbond" input file (see section 3 · 5 · 2), is allocated to such list. Tol_Num_MHbond, Tol_Num_SMHbond and Tol_Num_SSHbond are also defined in *hbond[ ]* and refer, respectively, to: the total number of inter-mainchain, sidechain — mainchain, and inter-sidechain, hydrogen bonds.

Cat	type	start				end			
		amino_num	virt_ amino_num	amino_type	at_ref	amino_num	virt_ amino_num	amino_type	at_ref
5ON	SIDE_MAIN_NH	1	1	T	4	3	3	Y	-1

FIGURE 5 · 2 Data Structure of HBOND Record

Note that the length information, group 2 in "x.hbond" file, is replaced by a *type* field. For instance, in the case of inter-mainchain hydrogen bonds, *type* is governed by R, the difference between amino acid sequence numbers of the CO and NH groups involved. This variable field is referred to by name, using Algorithm 5 · 1, as described below. This algorithm differentiates between sorts of hydrogen bond types (inter-mainchain as well as sidechain hydrogen bonds). Once all the hydrogen bonds are read in and the *type* fields processed, the whole *hbond[ ]* table is in an ordered state, having all inter-mainchain hydrogen bond class listed before the class of sidechain hydrogen bonds (which include sidechain — mainchain as well as inter-sidechain hydrogen bonds). Recall from Chapter 3 that, for every entry, the

start.amino_num sub-field value is always less or equal to the value of *end.amino_num* (ensuring that the direction of the chain goes from the N-terminus to the C-terminus). This facilitates bond searching when processing one or the other class, or category.

Note also that for each of the last two fields, *start* and *end*, used to gather information about the two residues involved in bonding, there exist two amino acid numbers: *amino_num* and *virt_amino_num*. Sometimes, amino acids are missing in the Brookhaven PDB although the chain is intact and, as a consequence, the numbering sequence is broken; this is because of an attempt to make aligned families of homologous proteins have the same numbering system. *virt_amino_num* (the virtual amino acid number) refers to amino acid sequence numbers for a continuous numbering of the sequence without disruption, while *amino_num* (the exact amino acid number) refers to the ones recorded in the PDB; often, these two numbers are identical. *amino_num* is needed to display labels with their real amino acid number while *virt_amino_num* is used to ease hydrogen bond searching in *hbond[ ]* and to generate the correct value of R.

Algorithm 5 · 1

To differentiate between hydrogen bond types, an algorithm is devised that stores in the *type* field the name values allocated. It is divided into three stages.

In the first stage, while the "x.hbond" file is read, the *type* field in *hbond[ ]* detects only:

- inter-mainchain hydrogen bond *types* without

specifying if they are part of a 'double bond', a parallel beta-sheet or an antiparallel beta-sheet.

- all types of the second category.

The second stage consists of ordering *hbond[ ]* so that inter-mainchain hydrogen bonds are listed first, followed by sidechain — mainchain hydrogen bonds.

In the last stage, a filtering process is performed while *hbond[ ]* is read again, and the *type* field is altered for those hydrogen bonds belonging to antiparallel beta sheets, parallel beta sheets or 'double bonds'.

As the *type* field is a variable referred to by name, I have followed a convention for naming hydrogen bonds. In Figure 5 · 3, all the hydrogen bond conventional names are listed with the corresponding value on their left.

Inter-mainchain hydrogen bonds have a name of the form:

MAIN_MAIN_x

where x is either 0, MIN1, PLUS1, MIN2,...,MIN5, PLUS5, MINBIG and PLUSBIG depending on R, the difference between amino acid sequence numbers of the CO group and NH groups involved (recall from Chapter 4 that R represents the distance as: the CO group of residue i binds to the NH group of residue i + R). These bonds are easily detected during the first stage of the algorithm.

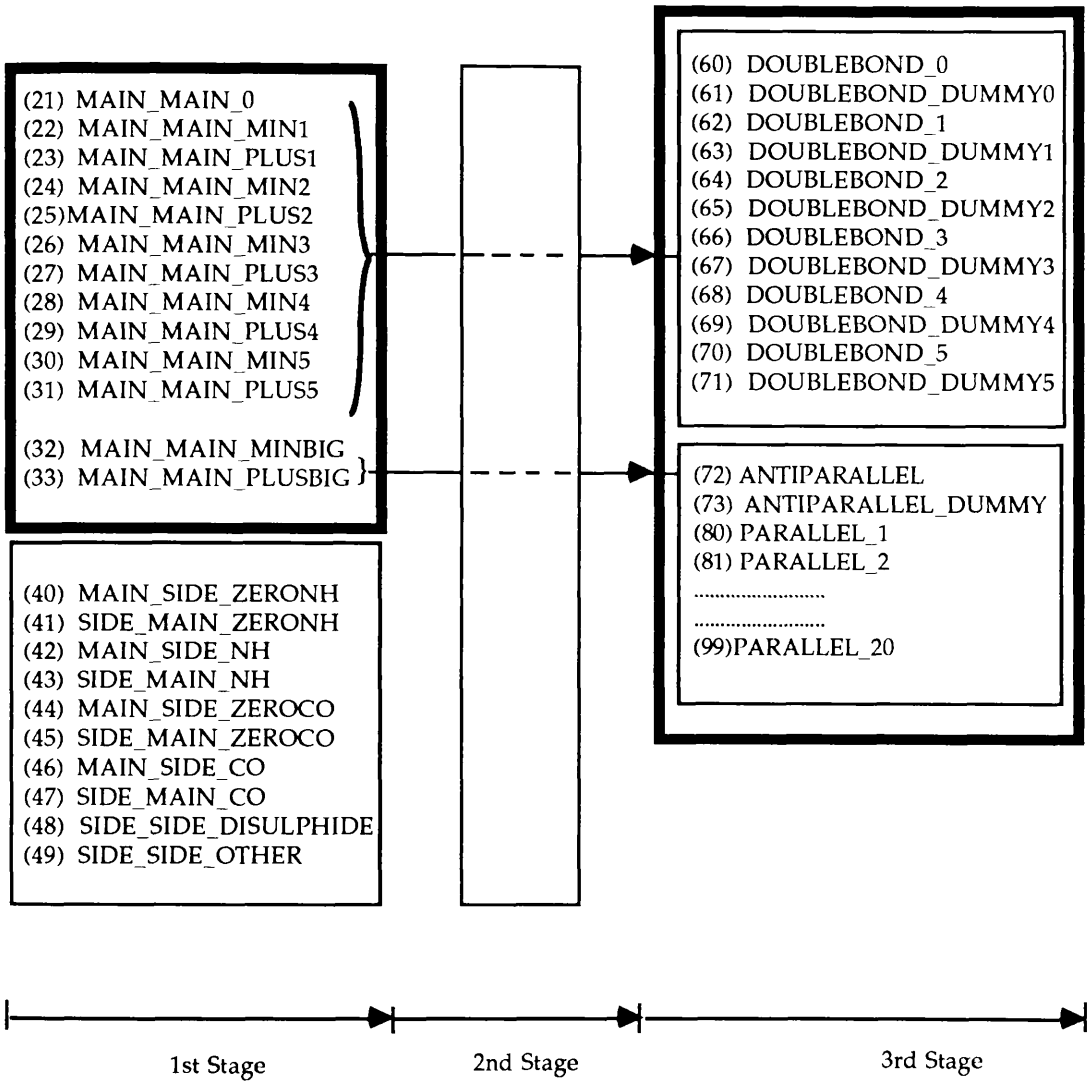


FIGURE 5 · 3 The Method of Determining Types of Hydrogen Bonds During Stages 1 and 3 of Algorithm 5 · 1

'Double bonds' are of the two forms:

DOUBLEBOND_x and DOUBLEBOND_DUMMYx

where x is 0, 1, 2, 3, 4 or 5, depending on the absolute value of R. Note that, for a 'double bond', either the CO->NH group or the NH->CO group is referred to as DOUBLEBOND_x , while the other group is a dummy (e.g. DOUBLEBOND_DUMMYx). These types of bonds are detected during the third stage of the algorithm.

If a 'double bond' has a value of x greater than 5 or less than - 5, it is called ANTIPARALLEL while its 'twin' is ANTIPARALLEL_DUMMY. This type of bond belongs to an antiparallel beta-sheet. They are detected during the last stage of the algorithm.

Hydrogen bonds that belong to parallel beta-sheets, have the name:

PARALLEL_x

where x denotes the pair of strands x (or ladder x) that encapsulates the 'parallel' hydrogen bond. $1 \leq x \leq 20$, thus allowing up to 20 different ladders to exist in a protein. During the last stage, these bonds are detected. Note that the value allocated to them (80 and over) is the highest compared to the other *types*, as this facilitates the filtering process at stage 3 of the algorithm (see Figure 5 · 3).

For sidechain — mainchain hydrogen bonds, names are allocated with the rule that direction from the N-terminus to the C-terminus must be specified. For example, MAIN_SIDE_NH represents an NH group of a main chain residue bonding to a side chain atom of a residue on the other side of the first residue with respect to the N-terminus. On the contrary, SIDE_MAIN_NH is used for an NH group bonding to a side chain atom of a residue at the N-terminal side of the NH group residue. The same holds for MAIN_SIDE_CO and SIDE_MAIN_CO. These specifications are needed to draw diagrammatic view representations. Note that MAIN_SIDE_ZERONH and SIDE_MAIN_ZERONH have the same meaning but are just differentiated to follow the logic of the notation. Disulphide bonds are named SIDE_SIDE_DISULPHIDE and all the inter-sidechain hydrogen bonds have SIDE_SIDE_OTHER as a name attached to them in the *type* field. All sidechain — mainchain hydrogen bonds as well as inter-sidechain bonds are easily found during stage 1 of the process.

While stage 1 and stage 2 of the algorithm are relatively easy to implement, stage 3 of the algorithm necessitates attention and is now described. Its code is listed in Figure 5 · 4.

It is during stage 3 that 'double bonds', bonds belonging to antiparallel beta sheets and bonds belonging to parallel beta sheets, are detected. For this, the *hbond[ ]* list is scanned again for all inter-mainchain hydrogen bonds; that is to say, only the first Tol_Num_MHbond entries of *hbond[ ]* are considered, starting with the first one on the list. To simplify this, I refer from now on to the two extremities of a hydrogen bond, or element, as the start and end residue.

Any element x in the list is first tested for 'double bonding' by checking neighbouring bonds with the same start residue. If one is

```
                    /************************************************/
                     /*****************Algorithm 5 - 1***************/
                    /************************************************/

#define     PARALLEL  80            /*Reference by name to first pair of strands, or ladder*/
...
...
...
Parallel = PARALLEL;                /* set initial value number for eligible ladder*/
for( b = hbond; b - hbond < Tol_Num_MHbond; b++)  {
    for( m = b + 1; m->start.virt_amino_num == b->start.virt_amino_num; m++)  {
        if ( m->end.virt_amino_num == b->end.virt_amino_num ) { /* check possible doublebond*/
            switch (m->type) {
                caseMAIN_MAIN_0:
                    b->type = DOUBLEBOND_0;
                    m->type= DOUBLEBOND_DUMMY0;
                    break;
                caseMAIN_MAIN_MIN1:
                caseMAIN_MAIN_PLUS1:
                    b->type = DOUBLEBOND_1;
                    m->type= DOUBLEBOND_DUMMY1;
                    break;
                case.....
                    ......
                caseMAIN_MAIN_MIN5:
                caseMAIN_MAIN_PLUS5:
                    b->type = DOUBLEBOND_5;
                    m->type= DOUBLEBOND_DUMMY5;
                    break;
                default :
                    b->type = ANTIPARALLEL;
                    m->type = ANTIPARALLEL_DUMMY;
                    break;
            }
        }
    }

    /* consider only bonds where residue difference is 5 or more */
    if ( (b->type != MAIN_MAIN_MINBIG) && (b->type != MAIN_MAIN_PLUSBIG) )
        continue;

    else                                    /* Check the possibility of a 'parallel' bond */
        if ( (find_beta(hbond + (b - hbond ) ))  /*recursive routine returning 1 if found  */
            b->type = Parallel++;           /* allocate proper type value before...*/
                                            /* ....referring to next ladder */
}
```

FIGURE 5 · 4 Code for Generating Inter-Mainchain Hydrogen Bond Types (Algorithm 5 · 1)

found with the same end residue as *x* (and thus is unique), the search ends for element *x* which is a 'double bond'. Its neighbour is allocated a 'dummy double bond' *type*, while *x* is allocated a 'double bond' *type* depending on its neighbour anterior *type* (found from stage 1). Note that, if the anterior *type* was MAIN_MAIN_PLUSBIG or MAIN_MAIN_MINBIG, *x* is part of an antiparallel beta sheet.

If there exists no such neighbour, *x* is tested for a possible 'parallel' bond only if its present *type* is MAIN_MAIN_PLUSBIG or MAIN_MAIN_MINBIG. Otherwise, the search ends for *x* and the next elements in the list are checked in turn.

To see if *x* is part of a particular pair of strands within a parallel beta sheet, a recursive routine is called. This routine not only checks if *x* is a 'parallel' bond but it finds, if *x* is a 'parallel' bond, all the other 'parallel' bonds that belong to the same pair of strands as *x*. The routine when called returns 1 if a ladder is found.

The routine starts by scanning the *hbond[ ]* table from *x*. Note that *x* is either a CO->NH bond connection or the other way around. The code for both alternatives is listed in Figure 5 · 5 and only the first case is detailed in what follows. Let us suppose that *x* is a CO–>NH bond connection from residue i to residue j (with i + 5 < j). Consideration is only given to neighbouring entries to *x* with a start residue between i and i + 2. There are now two possible chances for *x* to be a 'parallel' bond: either there exists a neighbour with a start residue two amino acids after *x* and the same end residue as *x* (this neighbour has obviously a NH->CO bond connection); or, if this case fails, there may be a neighbour which has its start and end residues equal respectively to i + 2 and j + 2 (this neighbour obviously has a CO->NH bond connection). If this

```
                    /************************************************/
                     /*************Algorithm 5 - 1 continued************/
                    /************************************************/


int  find_beta(b)              /*find all bonds belonging to same ladder, and allocate them...*/
/************** /              /*... the same ladder  value number */
HBOND    *b;
{
HBOND    *m;
if ( b->type == MAIN_MAIN_PLUSBIG) {      /* CO—>NH connection */
    for(m = b + 1; m->start.virt_amino_num <= b->start.virt_amino_num + 2; m++) {
        if ( (    ( m-> start.virt_amino_num - b->start.virt_amino_num == 2)
        &&        ( m->end.virt_amino_num - b->end.virt_amino_num ==0) )
        || (      ( m->start.virt_amino_num - b-> start.virt_amino_num == 2)
        &&        ( m->end.virt_amino_num  -b->end.virt_amino_num == 2) ) )
            {
            find_beta(b + (m - b) );          /* find rest of bonds belonging to same ladder */
            m->type = Parallel;               /* when done, allocate same ladder  value as b */
            return (1);
        }
    }
    return (0);                               /*  there is no more bond forming the same ladder    */
}

if ( b->type == MAIN_MAIN_MINBIG) {       /* NH—>CO connection */
    for(m = b + 1; m->start.virt_amino_num <= b->start.virt_amino_num + 2; m++) {
        if ( (    ( m-> start.virt_amino_num - b->start.virt_amino_num == 0)
        &&        ( m->end.virt_amino_num - b->end.virt_amino_num ==2) )
        || (      ( m->start.virt_amino_num - b-> start.virt_amino_num == 2)
        &&        ( m->end.virt_amino_num - b->end.virt_amino_num == 2) ) )
            {
            find_beta(b + (m - b) ) ;
            m->type = Parallel ;
            return (1) ;
        }
    }
    return (0) ;
}
}
```

FIGURE 5 · 5 Code of the Recursive Routine Detecting Bonds Belonging to the Same Pair of Strands (Algorithm 5 · 1 Continued)

neighbour is found, a recursive call to the same routine is applied for the neighbouring elements in order to find all bonds in the same ladder. The process is repeated until no such bonds are found anymore. Thereafter, the value number of the ladder is allocated to the *type* field of all these elected bonds, starting with the last and going up the 'tree' until x. At that time, the ladder is fully defined and new eligible 'parallel' beta-sheet bonds are checked with respect to the next ladder.

5 · 4 · 1 · 3 Dihedral Angle File

Finally, to store dihedral information from the input file "x.dihed", a linear list, *dihed[ ]*, is created. Each entry, DIHED, has the data structure shown in Figure 5 · 6. The three fields of DIHED structure store, respectively, the amino acid type (*amino_type* field), psi angle (*psi* field) and phi angle (*phi* field) needed to portray dihedral angles in diagrammatic pictures.

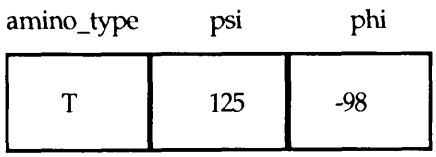


FIGURE 5 · 6 Data Structure of DIHED Record

5 · 4 · 2 Tramline Drawing Techniques

Since the main graphics primitives needed to visualize proteins are lines to draw line-segments, the heart of the package is concentrated

on three tables: *polypeptide_line[ ], diagrammatic_line[ ]* and *detail_line[ ].* Each one is devoted to its appropriate window. These 3 tables have the same data structure for consistency and ease of implementation of the various routines of the software in a modular fashion. Each of them contains the geometric and attributes of the line-segments to be drawn, organized to facilitate processing. Geometric information includes boundary coordinates and parameters to identify the spatial orientation of bonds and atoms. Attribute information includes designations of any colour, type and other relevant features to be applied to line-segments. The data structure of an entry of either table, called LINE, looks like the one in Figure 5 · 7.

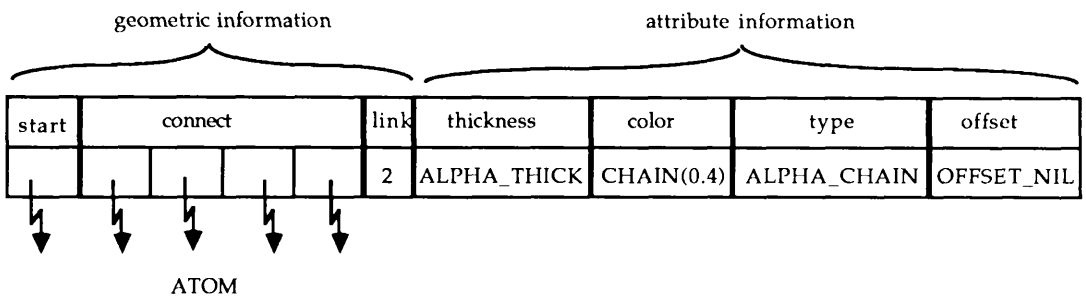


FIGURE 5 · 7 Data Structure of LINE Record

Geometric information in LINE can be organized in several ways. A convenient method to draw a line-segment would be just to specify its two end-point coordinates. This would mean having only two pointers to ATOM structures (one for each end-point) in LINE. However, with this data structure, tramlines in polypeptide chain pictures and atomic detail ones cannot be drawn easily. Consider what happens at the A end, in Figure 5 · 8, when tramlines are drawn between two consecutive line-segments AB and AC. Let us assume that depth of a line-segment is

taken as its midpoint and that AC is closer to the viewer. Hence, tramlines are drawn first for segment AB (Figure 5 · 8 · a). Then, the line-segment (rectangle) is coloured with an appropriate shade depending on its depth (Figure 5 · 8 · b). The same process is performed for line-segment AC (Figure 5 · 8 · c and Figure 5 · 8 · d). Note that in Figure 5 · 8 · d, one of the tramlines of AC cuts into the coloured portion of AB, producing a notch; the other tramline is short, producing a gap.

To alleviate this problem, a time-consuming algorithm was developed by Poet and Milner-White [POET 86]. However, this procedure is too slow for interactive rotation of a model, and some other implementation technique has to be devised to achieve interactive manipulation of pictures, while using tramlines depth perception.

Therefore, the major factor which led me organize LINE structures this way is due to tramlines. The new approach is illustrated in Figure 5 · 9 where line segments are drawn according to the following rule:

> All connecting bonds are drawn as line-segments that run from the atom in question to a point half way between the two atoms being joined. Thus, each bond consists of two segments, one contributed by each atom being joined.

Circular brushes are implemented to draw line-segments in order to ease tramline representation. Moreover, use of these brushes is not only helpful for tramline representation but is, by itself, an enhancement to three-dimensional perception, as already seen in section 4 · 6. The technicality of this method is now explained by an example (shown in Figure 5 · 9) where line-segments around atoms A and B are to be plotted.

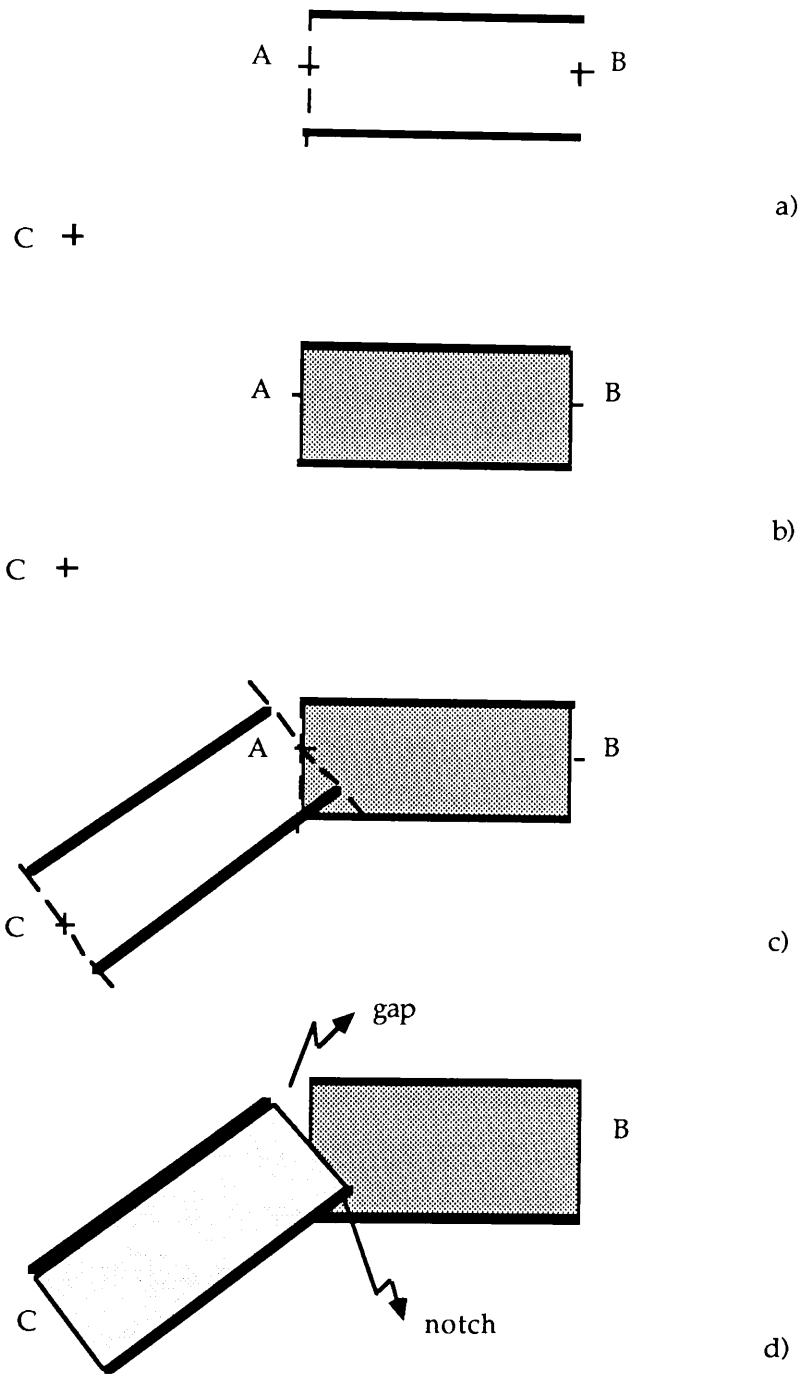


FIGURE 5 · 8 Drawing Tramlines when Line-Segments Directly Join Two Atoms

5 · 4 · 2 · 1 Tramlines Around Polypeptide Chains

In this example, A, B and C represent three consecutive smoothed alpha-carbons in the display of polypeptide chains with only inter-mainchain hydrogen bonds visualized. Examination of the procedure for the other pictures follows afterwards. As terminology, alpha-carbon segments (or chain segments) are referred to as line-segments going from one alpha-carbon towards the next in the polypeptide chain pictures; in the atomic detail pictures, line-segments, referred to as main chain segments, go from main chain atom towards the next atom; and side chain segments are to the ones departing from a side chain atom.

Referring to Figure 5 · 9 in which depth sorting is carried out for all line segments, let us assume that atom C is the closest to the viewer, then comes B, and finally A. In Figure 5 · 9 · a, a circle is first drawn in the background colour and centered at A; its radius is equal to that of the one used to plot alpha-carbon segments. Also, the circular brush used to draw this circle has a radius (the brush thickness) of 2 pixels. After this, tramlines are drawn in the background colour (with the same brush thickness as the one used to draw the circle) from points on the circle to points only half way between AB and AC. These tramlines are parallel to, and on each side of, the alpha-carbon segments. The next step is to draw the alpha-carbon segments AB and AC (see Figure 5 · 9 · b) according to the rule listed earlier. The brush thickness drawing alpha-carbon segments is predefined initially but can be altered by the user, before the start of a session, without recompiling the whole program. A variation of this value (expressed as a number of pixels) will make the polypeptide chain look either thicker or less thick. The shading used for the alpha-carbon segments around A depends on the depth of A, its z-coordinate. The same technique is performed

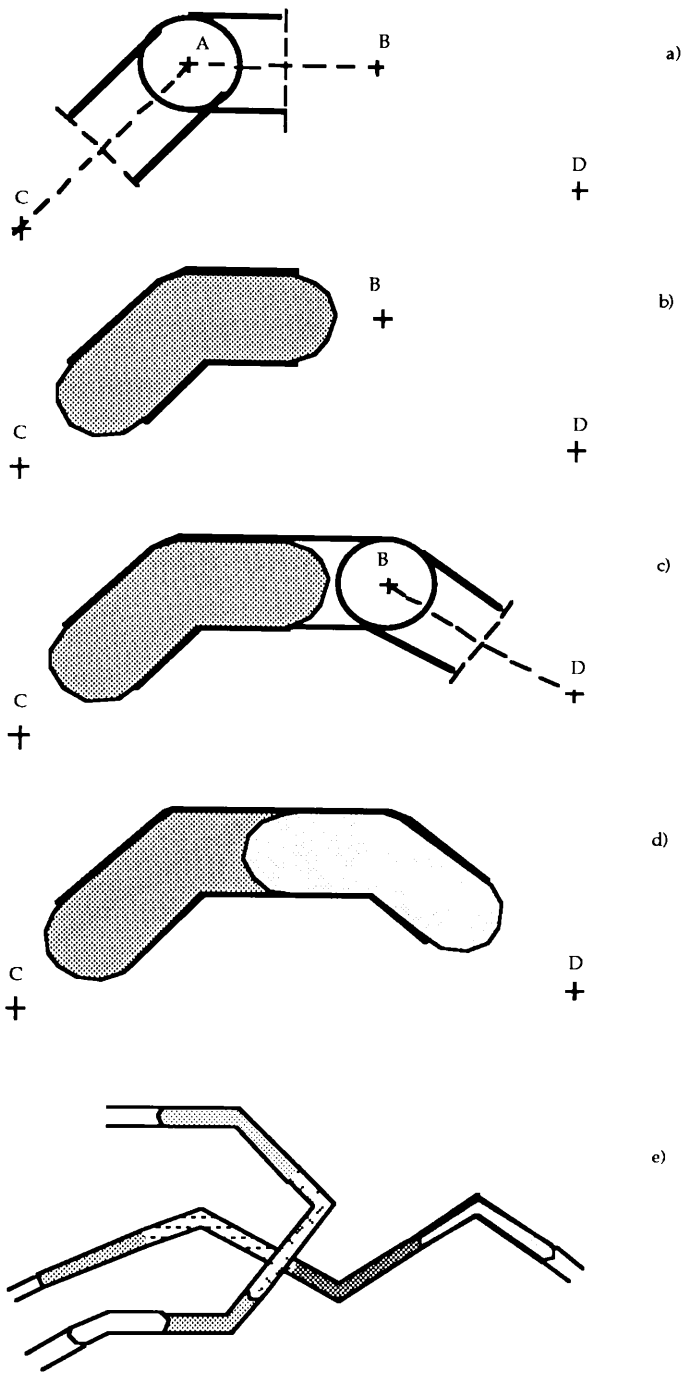


FIGURE 5 · 9 Drawing Tramlines Around Alpha-Carbon Segments

around atom B (Figure 5 · 9 · c and Figure 5 · 9 · d) which has two connecting bonds BA and BD. Note that the shading used is lighter as B is in front of A; this gives a depth cueing effect. This process is repeated for every atom until the whole protein is drawn (Figure 5 · 9 · e).

Note that, to draw tramlines, normal vectors to every connecting bond need to be computed. In Figure 5 · 10, the coordinates of the following points need to be computed in order to plot tramlines emerging from atom A: P1, P2, P3, P3, P4, R1, R2, R3 and R4. To do this, the normal vectors **p** and **r**, to bonds AB and AC, respectively, are computed, together with the half way points I and J between AB and AC.

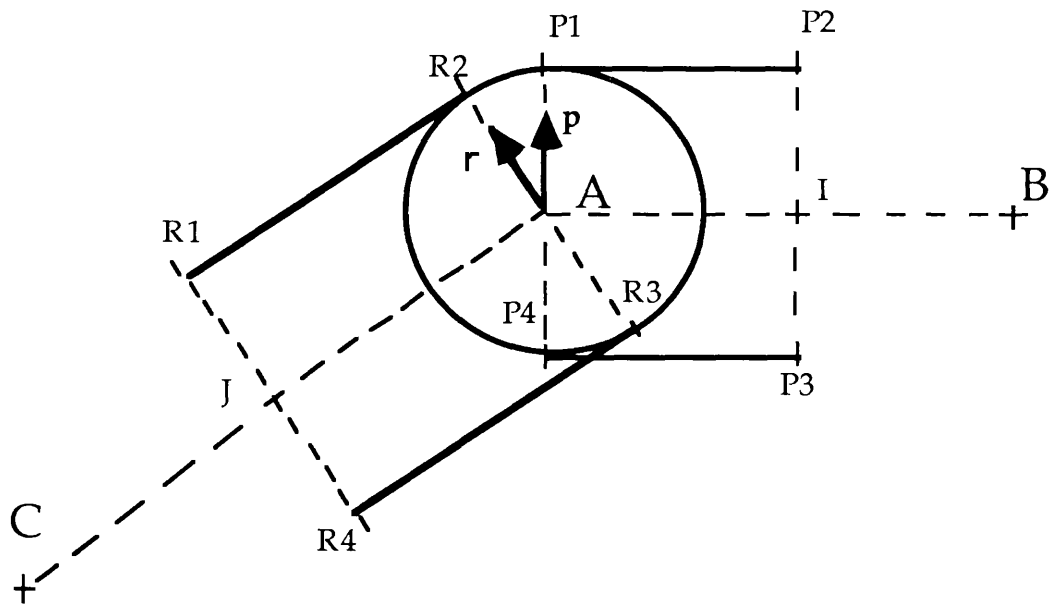


FIGURE 5 · 10 Boundary Points Needed to Draw Tramlines

Care must be taken when allocating brush thicknesses to alpha-carbon segments. As shown in Figure 5 · 11, a thickness of 2 pixels in alpha-carbon segments causes tramlines to cut into the dark coloured

segment, producing a notch of 1 pixel in both sides of the chain. However, this side effect disappears for values of thickness above 2 pixels.

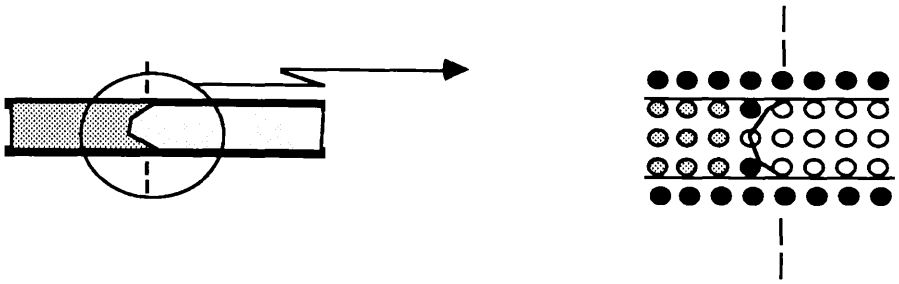


FIGURE 5 · 11 Appearance of Notches

Note that if antialiasing routines were implemented, which smooth out the lines, tramlines could be drawn using a brush thickness of only 1 pixel and at no time would a notch appear on the chain segments. But as this method is time-consuming, my procedure is better; it performs perfectly provided the chain segment thickness is above 2 pixels.

Some further explanation is needed to describe how tramlines are drawn around atoms in atomic detail pictures, and in polypeptide chain pictures displaying sidechain hydrogen bonds. With regard to the diagrammatic pictures, only the hydrogen bonds, portrayed by means of a pipe-like representation, need the use of tramlines. In these diagrams, tramlines are displayed by drawing hydrogen bond lines twice every time they are encountered, one on top of the other, but with two different circular brush sizes. The first line is drawn in the background colour with a thick line. The second is a redraw of the same line with a thinner circular brush size (one pixel less in thickness) and in the appropriate colour.

5 · 4 · 2 · 2 Tramlines around Tags

For the polypeptide chain pictures displaying sidechain hydrogen bonds, tramlines are drawn as illustrated in Figure 5 · 9, except for minor addition for certain bonds. It concerns those smoothed alpha-carbon atoms where side chains involved in hydrogen bonding are represented by tags. Recall from Chapter 4 that the direction of tags is computed by bisecting the obtuse angle formed by the smoothed alpha-carbon coordinates of the residue in question and those of its two neighbours. All tags have the same symbolic length of 1 Å. The way tramlines are drawn for such residues is illustrated in Figure 5 · 12, where alpha-carbon A is involved in bonding with side chain atoms.

First, the tramlines for the two alpha-carbon segments AB and AC are plotted as before (Figure 5 · 12 · a). Next, tramlines around the tag are drawn in the following way: a line is drawn, from A to a point 1Å from A, with a direction vector computed by bisecting the obtuse angle formed by C, A and B. The circular brush used has a thickness 1 pixel more than the ones used to draw alpha-carbon segments. It is drawn in the background colour (Figure 5 · 12 · b). This line is redrawn (Figure 5 · 12 · c) with the same brush thickness as the one for alpha-carbon segments; the shading depends on the position of A (its z coordinate).

Note that this technique avoids drawing a background circle and tramlines. It thus saves the computation time that would be needed to get tramline coordinate end-points. The final step is to draw the alpha-carbon segments AB and AC (Figure 5 · 12 · d) according to the rule given earlier. Note that the same shading is used for the alpha-carbon segments and the tag, as all three lines depart from A. Generally with

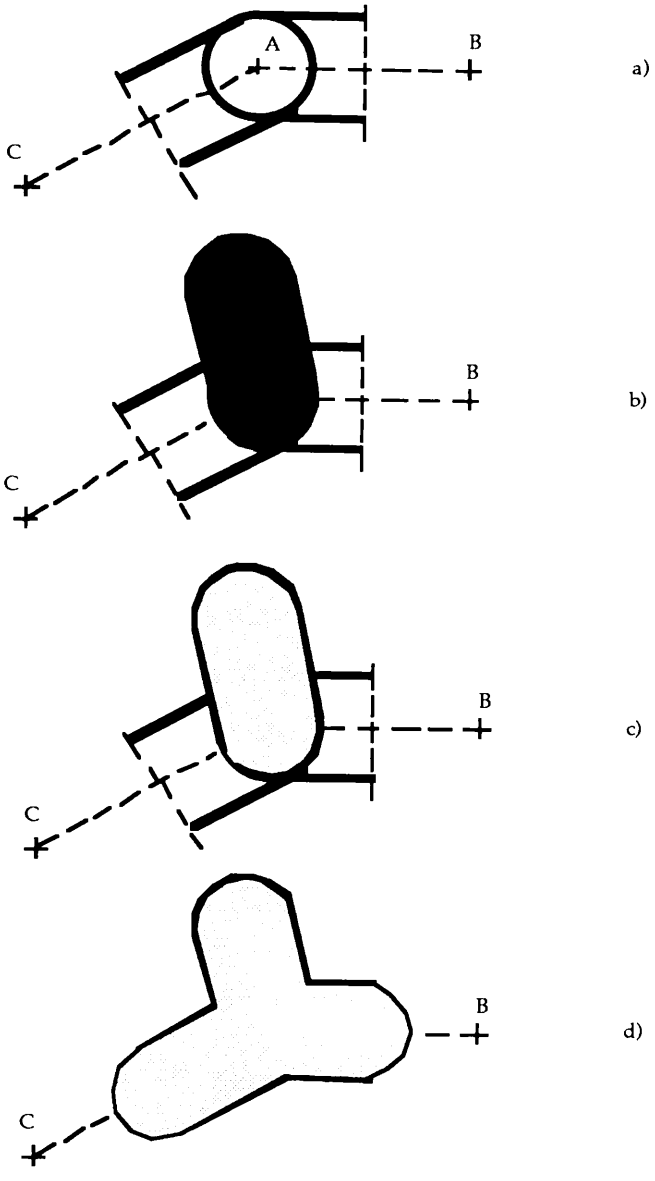


FIGURE 5 · 12 Drawing Tramlines Around Tags

this technique, notches do not exist.

5 · 4 · 2 · 3 Tramlines in Atomic Detail Pictures

In representing atomic details, a variation in brush thickness of 1 pixel between main chain segments and side chain segments allows one to differentiate which atoms belong to the polypeptide chain. This cue is in addition to their differentiation by colour (see section 4 · 6). Nevertheless, the same process as the one used for alpha-carbon segments is applied. When main chain atoms are drawn, the background circles used have a radius 1 pixel higher than the ones used for circles around side chain atoms. As before, the brush thickness used to draw circles and tramlines is fixed to 2 pixels. Figure 5 · 13 illustrates the way tramlines are drawn around an alpha-carbon C_α (where main chain bonds to side chain) in alanine. It has only one carbon side chain atom referred to as CB. The generalization of the technique for other residues as well as proline (which has covalent bond: nitrogen—side chain) is straightforward.

In Figure 5 · 13 · a, all main chain segments emerging from C_α are drawn together with their tramlines and background circle; the position of C_α is taken as being the closest to the viewer. In Figure 5 · 13 · b and Figure 5 · 13 · c, the side chain segment departing from CB is plotted. This is carried out after the circle and tramlines are displayed.

Zooming in at the boundary junction (see Figure 5 · 13 · c) shows some notches (3 in total) appearing on the Cα main chain segment. These notches might be thought unacceptable but, on the contrary, give a satisfying visual effect because they provide an articulated appearance at

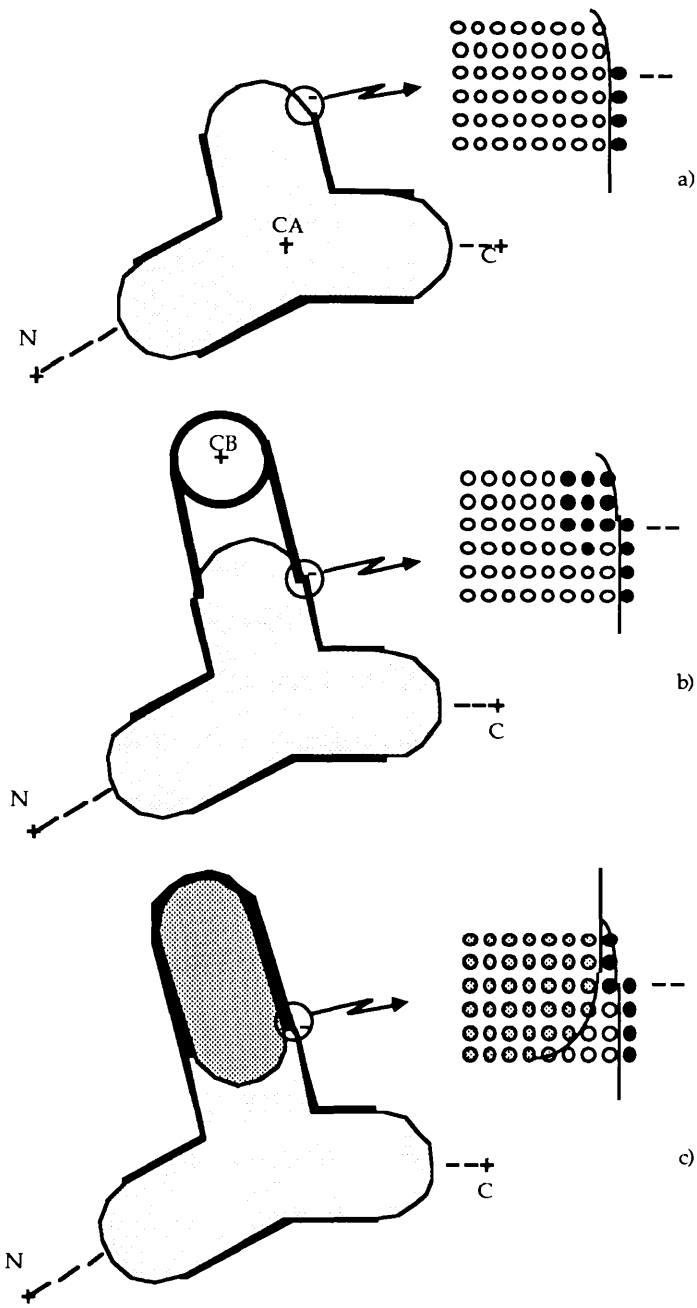


FIGURE 5 · 13 Drawing Tramlines Around Mainchain — Sidechain Connections

the junction. This is also depicted in the computer-generated picture shown in Figure 4 · 8.

For the alpha-carbon segments, the brush thickness for main chain and side chain segments can be altered before the start of a session. Note however, that the brush thickness used to draw main chain segments has to be 1 pixel higher than that for side chains.

It should now be clear how the LINE data structure is designed. The first field, a pointer to an ATOM record, indicates the position of the central atom where bond connections are to be made. The next field, containing four sub-fields all pointing to ATOM records, refer to its associate set of bond connections. Up to 4 atom connections are allowed for each atom since, in proteins, an atom cannot be bonded to more than 4 neighbouring ones. I adopted this LINE structure design for all three protein representations (polypeptide, atomic detail and diagrammatic view) where the different fields are described for each representation.

5 · 4 · 3 Polypeptide Chain Data Structure Design

For the polypeptide chain picture to be displayed, alpha-carbon information for the backbone chain of the protein under study are stored in an isolated table, *alpha[ ]*, having Tol_Num_Alpha as size (already specified in the "x.mol" input file). This information is copied directly from the *chain[ ]* table where each entry of *alpha[ ]* has the same data structure as ATOM. *alpha[ ]* also undergoes alteration due to positioning. It is performed for the alpha-carbon coordinates by applying the averaging method developed by Poet & Milner-White [POET 86] to smooth out the course of the chain (see Section 4 · 5 · 1).

Since the representation of polypeptide chains with side chains displays tags for those smoothed alpha-carbons whose side chains are involved in hydrogen bonding, a table, *tag[ ]*, is created, copying from *alpha[ ]* the information of the alpha-carbons in this category. The eligible alpha-carbons are found by scanning the *hbond[ ]* list and recording once only those whose side chains are bonded. ATOM structures are used, as well, for each entry. Although, as a rule, less than a third of the alpha-carbons are in this category, I preferred to allocate to *tag[ ]* half the *alpha[ ]* space (e.g. Tol_Num_Alpha / 2). Some preprocessing is made to the *coord* field in order to represent the 1Å tag distances.

The *polypeptide line[ ]* table consists of all the geometric and attribute information of the line segments to be drawn. This includes hydrogen bonds as well as smoothed alpha-carbons. LINE structures are used for each entry and the ordering of this table is organized in three layers: first, the chain segment information layer, then the inter-mainchain hydrogen bond information layer and finally the information layer for sidechain hydrogen bonds (see Figure 5 · 15). This is done in order to ease the searching process in the model. The block size allocation to *polypeptide_line[ ]*, Tol_Num_Polyline, is equal to the sum of Tol_Num_Alpha and Tol_Num_Hbond.

5 · 4 · 3 · 1 Geometric Information

Concerning the geometric information for alpha-carbons, the field *start* which is a pointer to an ATOM record in *alpha[ ]*, indicates the position of the center alpha-carbon where alpha-carbon connections are to be made. Among the four sub-fields of *connect*, the first two, also pointers to ATOM records in *alpha[ ]*, refer to its two neighbours. The

third sub-field refers to a tag position in the *tag[ ]* table in case of an alpha-carbon involvement in sidechain bonding; otherwise, it is ignored. The last sub-field is totally ignored for such representation.

Geometric information of all hydrogen bonds uses the *start* field and only the first sub-field of *connect*. These two pointers refer to the two extremities of the hydrogen bond line to be drawn. In order to get layer 2 and layer 3 reference pointers, *hbond[ ]* is scanned through its two *virt_amino_num* sub-entries. In case of layer 2 LINE entries, both point to ATOM entries in *alpha[ ]*; in layer 3, LINE entries where sidechain — mainchain hydrogen bond geometric information is stored, the *start* and *connect[0]* entries point to either *alpha[ ]* or *tag[ ]* but not both; in LINE entries belonging to layer 3, but where inter-sidechain hydrogen bonds are involved, both point to *tag[ ]*.

The *link* field is an integer variable that indicates the actual number of connections a line segment has. For polypeptide chain display with only inter-mainchain hydrogen bonds, this variable is set to 2 for all alpha-carbon segments. For sidechain hydrogen bond displays, this variable is altered to 3 for those alpha-carbons involved in side chain bonding. All hydrogen bond entries have a *link* field set to 2.

5 · 4 · 3 · 2 Attribute Information

thickness

The *thickness* field is an integer variable referring to the brush thickness that draws line-segments. Five different thicknesses of circular brushes are implemented although more can be added. The *thickness*

variable is the same for all alpha-carbon segments.

Generally, all hydrogen bonds have the same *thickness*. Exception is made for those inter-mainchain hydrogen bonds involved in 'double bonding' where both CO and NH groups of a pair of amino acids are bonded, and to the sidechain —mainchain ones where side chain bonds to a main chain atom of the same residue; disulphide bonds are part of this category too. The value of *thickness* for those hydrogen bonds is incremented by 1 compared to the other hydrogen bonds.

colour

The *colour* field specifies the colour and shade of line segments. A range of 8 gray intensity levels gives a reasonable indication of depth and are allocated to alpha-carbon segments. Note that choosing a red, green or blue colour gives a maximum of 3 intensity levels for either colour, as the actual video output of the CG1 has only a 6-bit output. Colour for alpha-carbon segments is allocated according to depth with nearer segments appearing in a brighter colour than more distant ones. Colours are referred to by names. Chain segment colour is referred to as CHAIN(x) where x, a real number ($0 < x < 1$), specifies the chain segment depth in relation to the range of chain depths of the molecule in question. The formula is as follows:

$$x = \frac{[(\text{z-max}) - (\text{z-depth})]}{\text{range}}$$

Notice that coordinates are with respect to a right-handed coordinate system. For example, with this formula and the way grey-scale intensity levels are specified in the lookup table (see COLOUR section in

appendix 3), the closest chain segment to the viewer is allocated the brightest intensity (e.g. CHAIN(1)).

Another point worth mentioning is that, it is important not to use a left-handed coordinate system as alpha-helices, for example, which in the biochemistry literature are always right-handed would become left-handed.

All hydrogen bonds are colour coded but colours indicate the type of bond rather than depth. A variety of different name colours is used (e.g. YELLOW, GREEN, etc...) rather than different shades of the same colour (e.g. CHAIN(x)).

type

The *type* field is an integer variable, referred to by name, which indicates the line segment type (either ALPHA_CHAIN segment or a variety of hydrogen bond types). All hydrogen bond types are listed in Figure 5 · 3. Every time the drawing process starts, layers 1 and 2 of this field are checked in order to be able to display either inter-mainchain hydrogen bonds or bonds involved in side chains.

offset

The *offset* field is a real variable that indicates an offset in the z-direction for the line segments drawn. It only affects hydrogen bonds. Its purpose is to avoid clutter when displaying alpha-helices as they contain many hydrogen bonds and chain segments. The value of the offset is fixed at 4.0 Å (e.g. referred to by the name OFFSET_HB).

5 · 4 · 3 · 3 Depth Sorting and Manipulation

For depth sorting of line segments for the polypeptide chain, a table *polypeptide_depth[ ]* is created, where each entry, DEPTH, has the data structure shown in Figure 5 · 14.

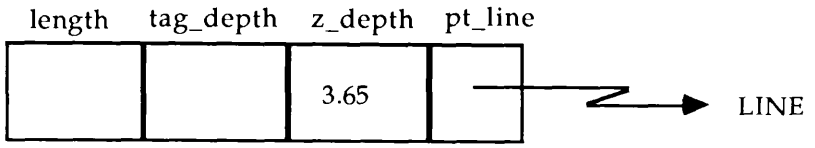


FIGURE 5 · 14 Data Structure of DEPTH Record

The table is allocated the sum of Tol_Num_Alpha and Tol_Num_Hbond as storage. Each DEPTH entry is linked to its appropriate LINE structure in *polypeptide_line[ ]* through the *pt_line* field. The *z_depth* field records the depth of line-segments; it involves fetching the corresponding z-coordinate(s), and an offset if appropriate. Note that depth of a hydrogen bond is taken as its midpoint added to the 4Å offset. Thus two z-coordinates are fetched, one from *start* and the other from *connect[0]*. The *tag_depth* and *length* fields are ignored and are used only for diagrammatic representations. Figure 5 · 15 shows the overall structure for the polypeptide chain model before depth sorting has been performed.

In order to display the picture, *polypeptide_depth[ ]* is depth-sorted through its *z_depth* field. An insertion sort seems adequate in this situation because few elements have to be switched at each stage since the change after a unit of rotation is quite small [COOK 80].

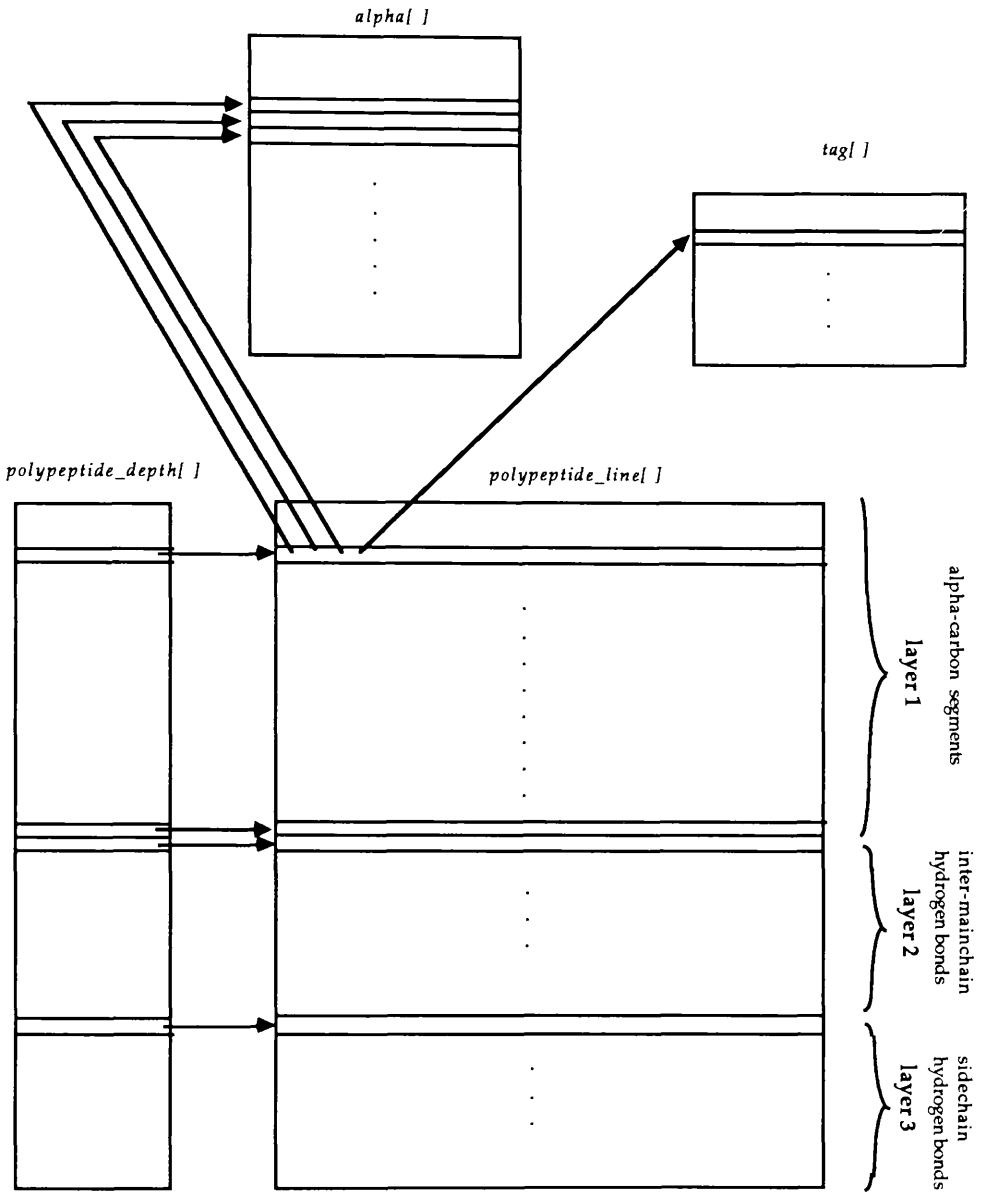


FIGURE 5 · 15 Overall Structure of the Polypeptide Chain Model Before the Depth Sorting Process

Switching between displays of inter-mainchain hydrogen bonds and of sidechain bonds, require one to update appropriately the *link* field of chain segments (a value of 3 to include tags, and 2 otherwise). After which, the drawing procedure is initiated in *polypeptide_depth[ ]* from top to bottom, with the further line segment being drawn first in this right-handed coordinate system. Note that in this procedure, the *type* field is always checked for every element in order to be able to display either type of bond.

For the display of fragments of the chain only, *polypeptide_depth[ ]* is re-initialized with the appropriate data. That is to say, all information in *polypeptide_depth[ ]* is destroyed and a new linking process is performed from *pt_line* to all three layers in *polypeptide_line[ ]* so that both their *start* and *connect[0]* fields refer to residues in the chosen fragments.

Manipulation of the model is performed by first updating *alpha[ ]* and *tag[ ]* accordingly. Note that if only fragments of the chain are displayed, only coordinates in their corresponding portion in *alpha[ ]* and *tag[ ]* are transformed. A procedure, described in section 5 · 5, ensures one that no overlaps exist between these portions.

A 4 x 4 concatenated transformation matrix, Transf_Update, recording all transformation operations done to the original polypeptide chain, is then updated. Going from the display of portions of the chain to the whole chain is done by re-initiating both *alpha[ ]* and *tag[ ]* and then applying a transformation using Transf_Update. This matrix is also needed for atomic detail pictures to display them initially in the right orientation. The *polypeptide_depth[ ]* table is then depth-sorted and the new depths for line segments are fetched. Figure 5 · 16 shows the

procedure, after performing the depth-sorting process. It has the same overall structure as of Figure 5 · 15. To indicate depth cueing for alpha-carbon segments, their *colour* entry in *polypeptide_line[ ]* is modified according to their new depth. Note that depth sorting and depth cueing changes are not required if the model is just scaled or translated.

5 · 4 · 4 Atomic Detail Data Structure Design

When atomic details of a segment of the polypeptide chain are chosen to be displayed, a table, *detail[ ]*, stores a copy of the portion of *chain[ ]* that corresponds to this segment. *detail[ ]* undergoes initial changes due to positioning (to fit the window display) and orientation (using the matrix Transf_Update so that atomic detail segments are in the right orientation). A storage space of 140 ATOM-structure is allocated to this table, allowing a maximum of 10 residues to be displayed (tryptophan is the 'longest' with 10 side chain atoms). More residues would clutter the display.

The *detail_line[ ]* table, similar to its 'parent' *polypeptide_line[ ]*, stores all the geometric and attribute information of the line segments to be drawn. The size allocated to *detail_line[ ]* is four times higher than the one of *detail[ ]* to suit any atomic detail fragment. The same information filing is used: atom information is stored first, followed by inter-mainchain and sidechain hydrogen bond information. The *start* and the relevant *connect* sub-fields point to ATOM structures in *detail[ ]*.

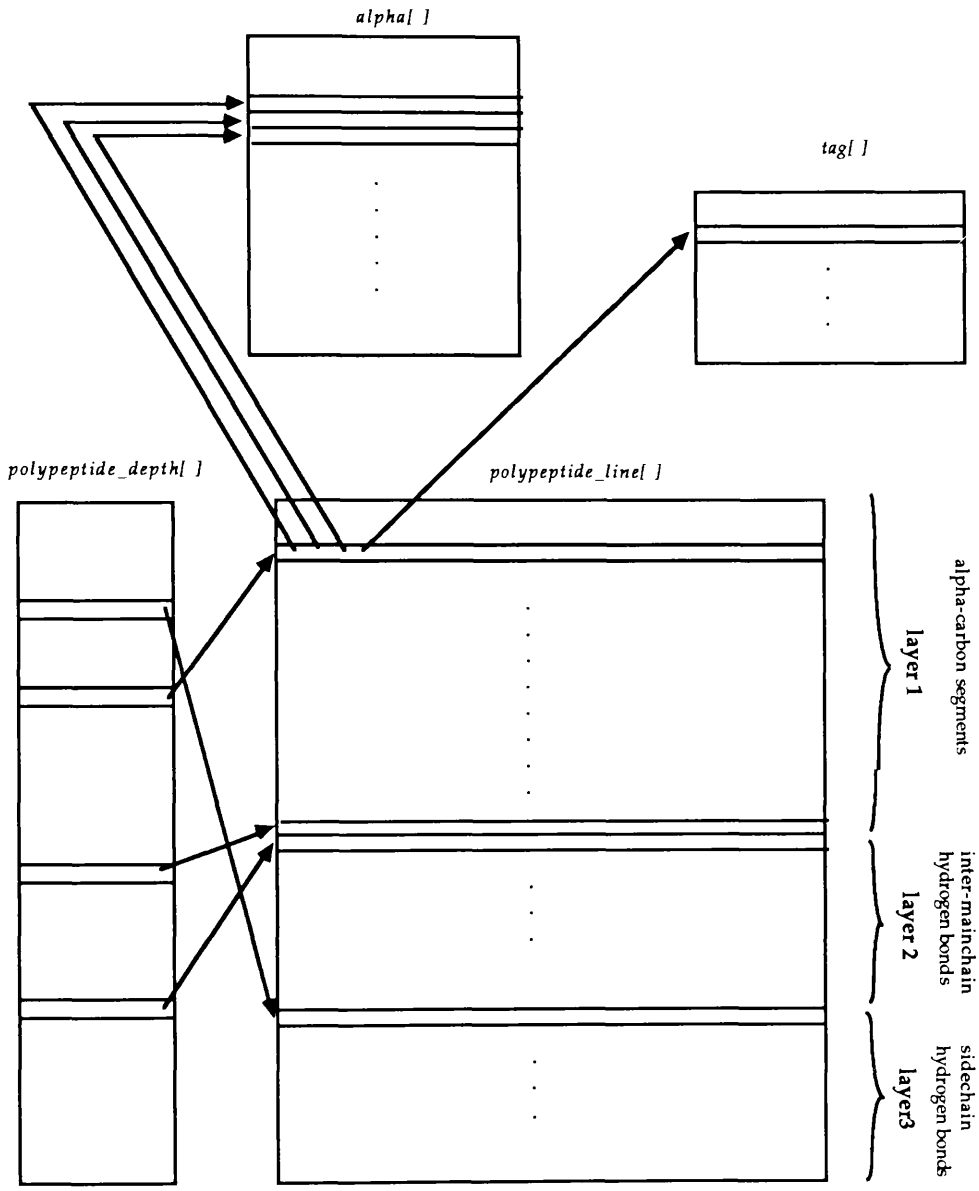


FIGURE 5 · 16 Overall Structure of the Polypeptide Chain Model After the Depth Sorting Process

5 · 4 · 4 · 1 Geometric Information

For information about atom geometry, the *start* pointer refers to the center atom where atom connections are to be made while the *connect* sub-fields point to these connecting atoms starting with main chain atoms. The reason is that two 'critical' atoms, C_α and occasionally N, bind to side chain atoms and the display of backbone chain atoms alone will be easily carried out without further search. Care should also be taken for the C and N atoms on the main chain involved in peptide bonds.

The field *link* indicates, as before, the actual number of connectors. Depending on whether all atoms constituting the portion of the chain or the ones forming the backbone of that portion of the chain are viewed, the *link* variable is updated for the 'critical' atoms.

Hydrogen bond geometric information uses, as before, the *start* field and only the first sub-field of *connect* to point to the two appropriate atoms in *detail[ ]* involved in bonding. In order to get layer 2 and 3 reference pointers, *hbond[ ]* is scanned through its *virt_amino_num* and *at_ref* sub-entries. All their *link* fields are set to 2.

5 · 4 · 4 · 2 Attribute Information

The *thickness* fields are allocated a value which maps the way line segments are drawn (see section 4 · 6).

Four different colours, with a range of 3 intensity levels each, are allocated to the main chain carbon atoms, side chain carbon atoms,

oxygen atoms and nitrogen atoms. The *colour* field, which holds these shades, refer to them by name respectively as MAIN_CARBON(x), SIDE_CARBON(x), OXYGEN(x) and NITROGEN(x) (for further details on how these intensity shades are specified in the lookup table, see section COLOUR in appendix 3). The variable x, as before, varies according to the depth.

All atoms to be drawn have a *type* field attached to them depending on their atomic type. Carbons in main chains and in side chains are distinguished. The hydrogen bond *type* fields are listed in Figure 5 · 3.

The offset field is disregarded throughout as no offset is added to the lines representing hydrogen bonds.

5 · 4 · 4 · 3 Depth Sorting and Manipulation

A depth-sorting table, *detail_line[ ]*, similar to its 'parent' is created and the same process as the one used for polypeptide chain representations is applied. Manipulation of the picture involves coordinate transformations of all atoms in *detail[ ]*.

5 · 4 · 5 Diagrammatic View Data Structure Design

Concerning the diagrammatic view of the protein under study, a table *diagram[ ]* similar to *alpha[ ]*, is created but with all its *coord* entries altered, to allow representation of the main chain as horizontal lines of boxes. While the *coord.x* and *coord.y* sub-fields vary, depending on the position of the 'boxes', all *coord.z* sub-fields are

ignored as no depth is considered. Two real variables X_Box and Y_Box are also defined in order to represent the width and hight of a box. Since initially the whole chain is displayed on the screen, X_Box has an initial value equal 1 (e.g. the ratio of the window width, fixed to Tol_Num_Alpha, by the total number of residues displayed in the row, Tol_Num_Alpha). Y_Box is controlled by the height of the character displayed as a residue label inside the box.

All geometric and attribute information of the line-segments to be drawn is stored in the table *diagram_line[ ]*, similar in structure to *polypeptide_line[ ]*. Since only hydrogen bond line-segments are displayed, the space allocated to this table is equal to Tol_Num_Hbond.

diagram_line[ ] has two layers corresponding to the 2 categories of hydrogen bonds. The *start* field and the first *connect* sub-field of every entry of *diagram_line[ ]* point to the two appropriate residues in *diagram[ ]* involved in bonding.

To get bonds involved in hydrogen bonding (e.g. the two reference pointers to *diagram[ ]*), a *type* check is made for every HBOND entry during the scanning process through *hbond[ ]*. This control is needed to record the 'sequence difference' (distance apart in sequence of the two residues joined) of every first 'parallel' bond of any ladder encountered. The information is stored in an vector table, Strand_Width, having the size of the maximum number of ladders allowed(e.g. 20). The table also records the number of 'parallel' bonds involved in each ladder. All this information is needed to draw the long rectangular boxes that portray ladders.

Attribute information for *diagram_line[ ]* is identical to that

allocated to *polypeptide_line[ ]*, except for the *offset* field. Since inter-mainchain hydrogen bonds are drawn at one side or the other of boxes to represent NH or CO groups (except for 'double bonds' which are drawn in the middle), the *offset* field is used to represent this small shift from the center of the box. This shift is measured with respect to the X_Box width of a box. The left-hand side of a box corresponds to the NH part while the right-hand side corresponds to the CO part.

For a hydrogen bond with a NH–>CO group connection, the *offset* is fixed to OFFSET_NH (e.g. reference by name to – 1/4); for a CO->NH connection it is OFFSET_CO (e.g. 1/4) and for 'double bonds' it is OFFSET_NIL (e.g. 0.0) All bonds in layer 3 are allocated OFFSET_NIL in their *offset* field as no shift is required. Note also that the *offset* field of any 'parallel' bond is set either to OFFSET_NH or OFFSET_CO depending on its *Cat* entry in *hbond[ ]* (e.g. 4NO => OFFSET_NH and 4ON => OFFSET_CO).

To display diagrammatic views, one more table is created: *diagram_depth[ ]*, similar in structure to the *polypeptide_depth[ ]* table. In this table, depth sorting of bonds is performed mainly according to their 'sequence difference', using the *length* field introduced in section 5 · 4 · 3 · 3. However, since different techniques are used for either inter-mainchain hydrogen bonds (category 1) or bonds involving side chains (category 2), *diagram_depth[ ]* is replaced by splitting it into two different tables: *diagram_main_depth[ ]* (with the size of Tol_Num_MHbond) and *diagram_side_depth[ ]* (with a size equal to the sum of Tol_Num_SMHbond and Tol_Num_SSHbond).

Linkage from *diagram_main_depth[ ]* (*diagram_side_depth[ ]*) to *diagram_line[ ]* layer 2 (layer 3) is performed appropriately through the *pt_line* pointers. Let us now consider the two cases separately.

5 · 4 · 5 · 1 Inter-Mainchain Hydrogen Bonds

For these displays, only the *length* and *z_depth* fields of *diagram_main_depth[ ]* are used. Since hydrogen bonds are to be drawn in depth order according to their distance apart in sequence, the *length* field is used.

The *length* field does not represent the exact 'sequence difference' for all bonds. A slight modification is performed on to all 'parallel' bond *length* in order to portray ladders as rectangular boxes (a more appropriate name for this field would be *virtual_length*). The algorithm that follows describes how *diagram_main_depth[ ]* is restructured in order to ease the rectangular representation of ladders.

Algorithm 5 · 2

When *diagram_main_depth[ ]* is initialized and the linking process between it and *diagram_line[ ]* starts, the *length* field records the 'sequence difference' of all bonds except that of 'parallel' bonds. For these particular bonds, the appropriate 'sequence difference' information stored in Strand_Width is allocated to their *length* field. This is mainly done in order to gather, in contiguous storage, all 'parallel' bond DEPTH entries belonging to the same ladder, once the depth sorting process is finished.

When the linking process between *diagram_main_depth[ ]* and *diagram_line[ ]* is terminated, a first sort is performed to *diagram_main_depth[ ]* in ascending order with respect to *length*.

Now that bonds are sorted, there exists a chance that intruder bonds

(either MAIN_MAIN_PLUSBIG, MAIN_MAIN_MINBIG or even 'parallel' bond types) are still 'infiltrated' between bonds belonging to the same pair of strands. In order to gather all DEPTH entries in contiguous storage for every ladder, the *z_depth* field is used to perform a local depth sort for every region where 'parallel' bonds belong to the same ladder.

The process starts by scanning the *diagram_main_depth[ ]* table from top to bottom. As soon as a 'parallel' bond is found, all intruder bonds found situated between it and its 'family' strand bonds are pushed down the table while the family bonds are moved up. The search resumes just after the last family bond entry, and the process goes on until the list is exhausted. The code for this local search/sort technique is listed in Figure 5 · 17.

Algorithm 5 · 3

Before the drawing process can start, a final adjustment is made to the *z_depth* fields in order to be able to display pipe-like hydrogen bonds without overlapping one another horizontally. Vertical overlaps are mostly avoided by drawing the longest pipes first and by means of the tramlines around bonds.

The technique devised uses the *z_depth* field to record the horizontal 'level' in which the corresponding line is to be drawn. Algorithm 5 · 3 ensures that the same horizontal level is not allocated to bonds where the residue sequence between the two connecting amino acids of one overlap the residue sequence of another. Another more time-consuming algorithm is also suggested later, but with it, hydrogen

```
                    /*************************************************/
                     /*****************Algorithm 5 - 2**************/
                    /*************************************************/


#define PARALLEL        80   /*  reference by name to first pair of strands, or ladder    */
.....


partial_depth_sort(depth, ndepth)
/*********************************/
DEPTH       *depth;
int         ndepth;
{
DEPTH       *d, *m, *r;
int         nbeta;      /*number of 'parallel' bonds belonging to a particular ladder */
int         nlocal;     /* number of bonds locally to be sorted*/

for(d = depth; d - depth < ndepth;)                 {
    if (d-> pt_line->type  < PARALLEL)              /*don't consider if not a 'parallel' bond */
        d++;
    else                                            {
        d-> z_depth = 0.0;      /*      force it to 0.0 as a 'parallel' bond intruder in an old....*/
                                /*....local sort is correctly set for this new one */
        for( m = d + 1; nbeta = nlocal = 1; m->length == d->length; m++, nlocal++)  {
            if ( m->pt_line->type == d-> pt_line->type ) {
                nbeta++;
                m->z_depth = 0.0;       /* lowest value to 'parallel' bonds of same ladder*/
            }
            else
                m->z_depth = 1.0;       /*higher value to all intruders */
        }
        r = depth + (d - depth);        /* reference to first 'parallel' bond  of ladder */
        depth_sort_strand(r, nlocal);   /* start local insertion sort */
        d += nbeta;                     /*skip all 'parallel' bonds of actual ladder*/
    }
}


depth_sort_strand(dp, ndp)
/*************************/
DEPTH       *dp;
int         ndp;
{
DEPTH       temp, *m, *p;

for(p = dp; p - dp < ndp; p++) {
    temp = *p;
    for( m = p - 1; m >= dp && tempz_depth < m-> z_depth; m--)
        *(m + 1) = *m;
    *(m + 1) = temp;
}
}
```

FIGURE 5 · 17 Code of the Local Sort for Parallel Bonds Belonging to the Same Pair of Strands (Algorithm 5 · 2)

bonds are displayed in a less informative way as compared to with Algorithm 5 · 3.

In order to apply Algorithm 5 · 3, a linear integer vector list *chainette[ ]* is created, having a size of twice Tol_Num_Alpha. The reason for having this size is to differentiate between the left-hand side of a box (NH group) and its right one (CO group). Cases occur when the end-residue of one bond (the bond being a CO->NH connection) and the start residue of another bond (the bond being also a CO->NH connection) emerge from the same amino acid, and consequently, one can draw both at the same horizontal level, as they would not overlap.

Initially, *chainette[ ]* has all its elements set to zero where each odd (even) element maps to its corresponding left-hand (right-hand) side box.

diagram_main_depth[ ] is processed from top to bottom, starting with the first entry (e.g. one that has the shortest bond *length*). For every entry of *diagram_main_depth[ ]*, the start residue and the end residue of a bond, as well as its *type*, are fetched from *diagram_line[ ]*. Then, all (contiguous) elements of *chainette[ ]* between the corresponding start residue and end residue of the bond are checked in order for a level to be allocated to that bond. The highest of those vector elements, incremented by one, gives the corresponding bond level (and thus *z_depth*) and all the vector elements bracketed by the bond in chainette[ ] are altered to the same level value before the next bond is processed. The code for Algorithm 5 · 3 is listed in Figure 5 · 18.

```
                    /*************************************************/
                     /*****************Algorithm 5 - 3***************/
                    /*************************************************/

#define     ARRAY(length,type)      (type *) calloc(length, sizeof(type) )
#define     OFFSET_NH               -0.25
#define     OFFSET_CO                0.25
#define     OFFSET_NIL               0.0

diagram_main_algorithm(depth, ndepth,Tol_Num_Alpha)
/*******************************************************/
DEPTH       *depth;
int         ndepth, Tol_Num_Alpha;
{
DEPTH       *d;
int         *x, *w, maximum, *chainette;
LINE        *l;

chainette = ARRAY(2*Tol_Num_Alpha, int);            /* allocate dynamically */
for(d = depth; d - depth < ndepth; d++)  {
    l = d-> pt_line;

    if ( (l-> type == DOUBLEBOND_DUMMY0)        /* ignore all 'dummy double bonds'*/
    || (l-> type == DOUBLEBOND_DUMMY1)
    || (l-> type == DOUBLEBOND_DUMMY2)
    || (l-> type == DOUBLEBOND_DUMMY3)
    || (l-> type == DOUBLEBOND_DUMMY4)
    || (l-> type == DOUBLEBOND_DUMMY5) )
        continue;

    if (l-> offset == OFFSET_NH)                        /* NH->CO group */
        x = chainette + (2 * (l-> start->amino_num       - 2);/* refer to left-hand side of a box */
        w = chainette + (2 * (l-> connect[0]->amino_num - 1);  /* "  right-hand side of a box*/
    }

    else if((l-> offset == OFFSET_CO) || (l->offset == OFFSET_NIL)) {/*include doublebds*/
        x = chainette + (2 * (l-> start->amino_num )      - 1);
        w = chainette + (2 * (l-> connect[0]->amino_num) - 2);
    }

    maximum = max_level (x, w) + 1;
    for(; w - x >= 0; x++)
        *x = maximum;
    d->z_depth = (float)(maximum);
}
cfree (chainette);
}

int       max_level (x,w)
/***********************/
int         *x, *w;
{
int         max, *i;

max = *x;
for( i = x; w - i >= 0; i++)  {
    if (*i > max)
        max = *i;
}
return ( max ) ;
}
```

FIGURE 5 ·18 Code for Generating Hydrogen Bond Levels (Algorithm 5 · 3)

Now, we are at a stage to draw the picture, starting with the last entry in *diagram_main_depth[ ]*. In order to represent tramlines, just before a 'pipe-like' representation of a bond is drawn with its appropriate *colour* and brush *thickness*, the same pipe is drawn in the background colour with a thicker brush. The technique for 'parallel' bonds is slightly different. First, a rectangular box is drawn at the center of the 'parallel' bonds in the background colour using a thick brush (Figure 5 · 19 · a). The rectangle length and height depend on, respectively, the length separating the pair of strands and the number of bonds involved in them. These are determined by checking the matrix Strand_Width, previously defined. Then follows the drawing of 'parallel' bonds using the same tramline technique as for the other pipe representations. However, the 'parallel' bonds are represented as pipes that are broken at their center, leaving a space that just fits the rectangular box (see Figure 5 · 19 · b and Figure 5 · 19 · c). Finally, the rectangular box is redrawn in its appropriate colour (see Figure 5 · 19 · d). This drawing sequence avoids insertion of any notches around the rectangle.

Variant to Algorithm 5 · 3

An alternative algorithm that follows the same rule (e.g. the same horizontal level is not allocated to bonds where the residue sequence between the two connecting amino acids of one overlap the residue sequence of another) as the one just described, is now suggested.

However, it was not adopted in my model, due mainly to two reasons. First, it is a time-consuming procedure compared to the one adopted; secondly, the display of secondary features such as alpha-helices are not as informative as those using Algorithm 5 · 3.

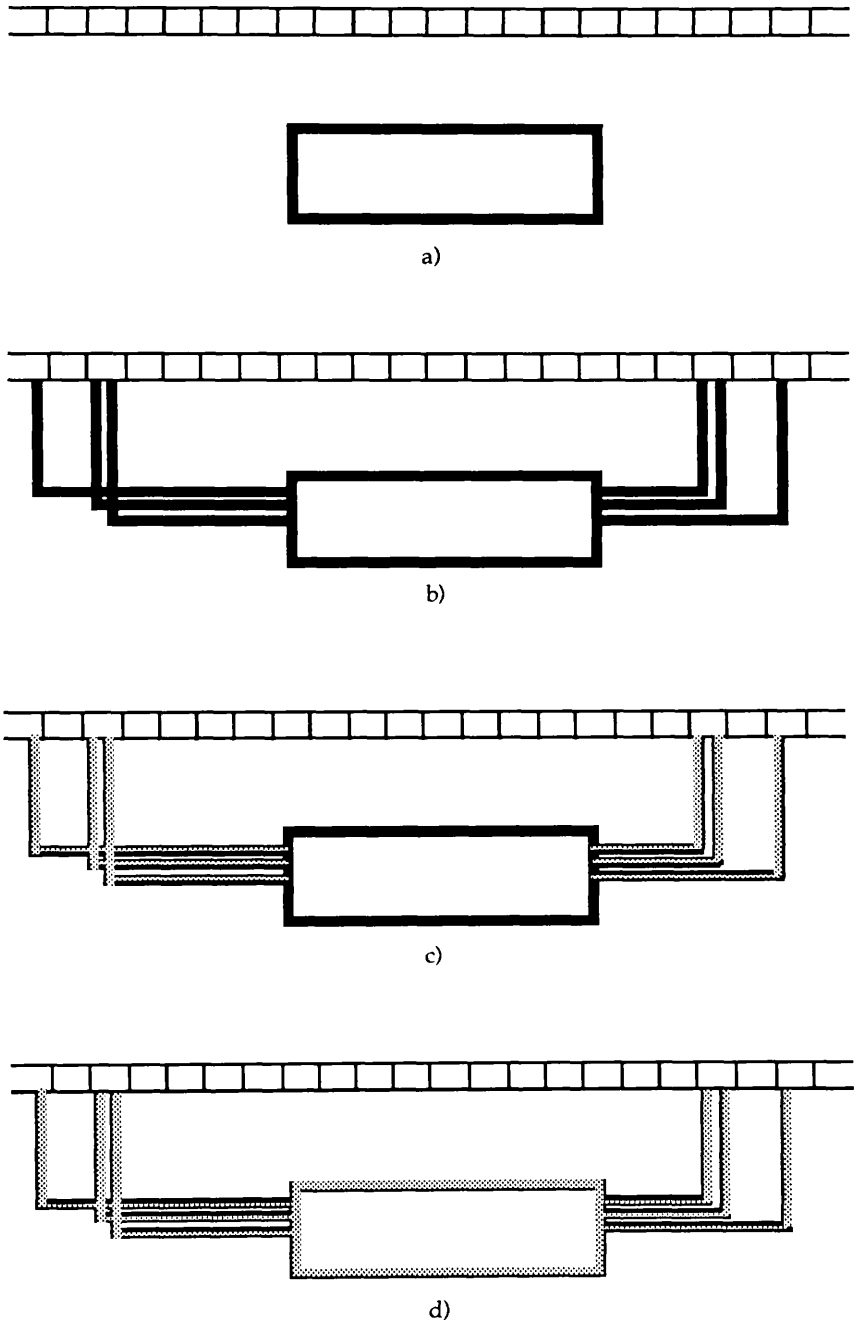


FIGURE 5 · 19 How to Draw Tramlines Around a Pair of Strands

Both algorithms allocate levels for bonds from the sorted *diagram_main_depth[ ]* table starting with the shortest and nearest bond with respect to the N-terminus. But while Algorithm 5 · 3 considers a higher level allocation for a particular bond, say *Bl*, with respect to all previously allocated bonds in the same overlapping region, this algorithm concentrates on locally considering all overlapping bonds having the same 'sequence difference' as *Bl*, and allocating to *Bl* a level that can be either higher or lower than its preceding one with the same 'sequence difference' and in the same overlapping region. Figure 5 · 20 illustrates how each algorithm tackles the problem of drawing bond *Bl* that belongs to an alpha-helix.

In the variant algorithm, the same vector *chainette[ ]* (defined as before) is created as well as a 'booking' list which is maintained for all bonds having the same 'sequence difference'. To get such a booking list, bearing in mind that the *diagram_main_depth[ ]* list is already sorted according to *length*, pointers into this list define the booking list of bonds for a current 'sequence difference'. As we move to the next 'sequence difference', the pointers are updated to define a new booking list.

In the booking list another list is maintained, called an active list, of bonds currently overlapping each other as the sequence is traversed from the N-terminus to the C-terminus. Initially, the active list has both its pointers (*active_start* and *active_end*) pointing to the start of the booking list. Suppose that, at a particular moment, the active list has the following bonds: *Bh*, *Bi*, *Bj* and *Bk*. As the next bond *Bl* is considered (with the same 'sequence difference'), the active list is first checked to see if all its bond elements overlap with *Bl*. All the non-overlapping bonds are taken from the active list one after the other and, simultaneously,

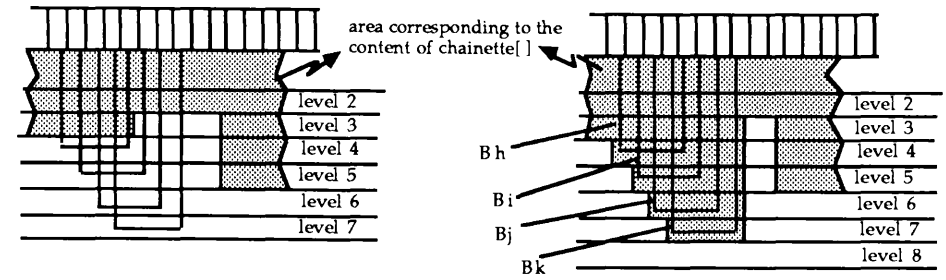
their corresponding portion in *chainette[ ]* is updated; by updating, I mean the alteration of the contents of those elements in *chainette[ ]*, between the corresponding start and end residues of a non-overlapping bond, to the level value of the bond just leaving the active list. Notice that eligible non-overlapping elements are contiguously allocated in *diagram_main_depth[ ]* and that the first one is pointed to by *active_start*.

Once the entire updating process is performed, and say *Bh* is the only non-overlapping bond, the updated active list becomes: $\{Bi, Bj, B_k\}$.

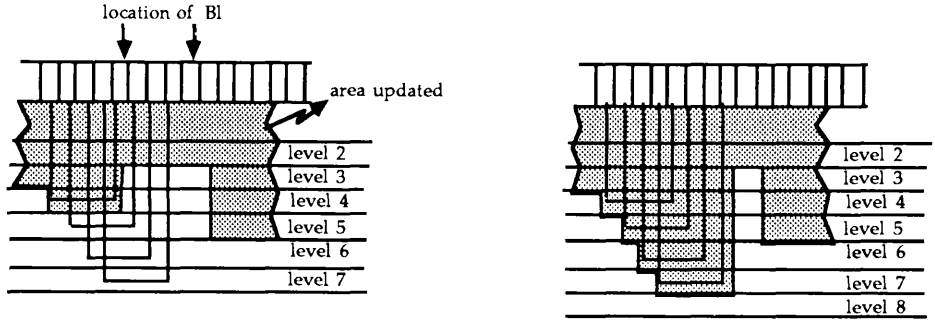
To determine the level for *Bl*, first the maximum content *max* of those elements in *chainette[ ]* between the corresponding start and end residues of *Bl* is obtained. Then, *max* is put on a temporary integer list, *temp[ ]*, to be sorted in ascending order with all the level values allocated to bonds of the updated active list. Note here that the level value of *Bl* should be higher than *max* so that this bond can be drawn without cluttering any shorter ones in the same region.

The level of *Bl* is found by comparing the difference of contents between two successive entries in the sorted list *temp[ ]* starting at the location of *max*, onward. As soon as the first difference greater than 1 is found (say between *temp[i]* and *temp[j]*), *Bl* is allocated the value of *temp[i]* incremented by 1. Otherwise, *Bl* is allocated the highest value of the *temp[ ]* table incremented by 1.

To illustrate the variant algorithm and by always referring to the left-hand side of Figure $5 \cdot 20$, the active list *A*, before bond *Bl* is considered, looks like $A = \{Bh, Bi, Bj, Bk\}$ (see Figure $5 \cdot 20 \cdot$ a). When *Bl* is considered for entry to the active list *A*, *Bh* leaves list *A* since it does not overlap with *Bl*, and *chainette[ ]* is updated accordingly (see the

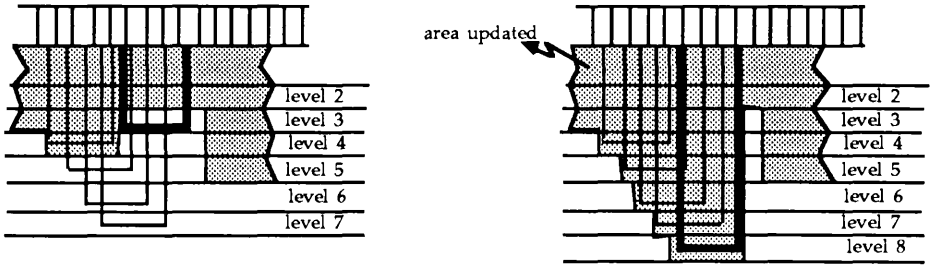


a) Before Bl is considered



Bh leaves the active list and the
shaded area is altered accordingly

b) When Bl is considered



c) When Bl is allocated a level

Variant of Algorithm 5 · 3

Algorithm 5 · 3

FIGURE 5 · 20 Strategy Adopted by Algorithm 5 · 3 and its Variant to Allocate a Level for a Hydrogen Bond

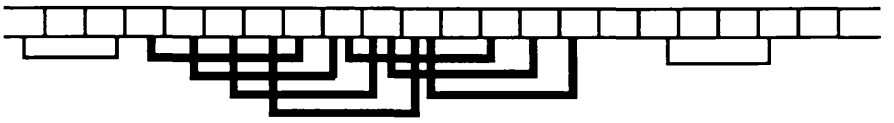
shaded region in Figure 5 · 20 · b). To get a level for *Bl*, all level values for the bonds in *A* (e.g. 4, 5 and 6) are inserted into table *temp[ ]*, together with *max* (e.g. 2). *temp[ ]* looks like *temp[ ]* = [2, 4, 5, 6] and is already sorted. The first check between the first element and the second (e.g. 4 - 2 > 1) is conclusive and *Bl* is allocated level 3 (e.g. 2 + 1).

Comparison between Algorithm 5 · 3 and its Variant

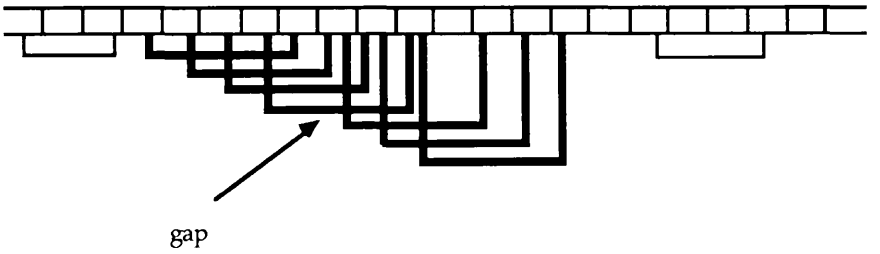
The main difference between the strategy adopted by Algorithm 5 · 3 and its variant is in the way the *chainette[ ]* table is updated. In Algorithm 5 · 3, *chainette[ ]* is updated as soon as a bond is allocated a level (see the right-hand side of Figure 5 · 20 · c), while, in the variant algorithm, *chainette[ ]* is updated only after a bond leaves the active list (see left-hand side of Figure 5 · 20 · b), even though a level is already allocated to it.

Since overlapping line segments are dealt with differently in the two algorithms, the way an alpha-helix is displayed might look different. The same alpha-helix is shown in Figure 5 · 21 · a, applying the variant algorithm, and in Figure 5 · 21 · b with Algorithm 5 · 3 applied. Note how a unique large staircase is formed using Algorithm 5 · 3. On the other hand, several small staircases, following each other, are displayed using the variant algorithm, making it difficult to know if all bonds belong to the same alpha-helix.

Also, from the two figures, it is evident that it is easier to notice any missing bond within an alpha helix from Figure 5 · 21 · b (shown as a broken stair) than from Figure 5 · 21 · a. It is quite clear that display of secondary structure features such as alpha-helices using the variant algorithm are not as informative as using Algorithm 5 · 3.



a) The missing bond not detected with the variant of Algorithm 5 · 3



b) The missing bond detected with Algorithm 5 · 3

FIGURE 5 · 21 Display of an Alpha-Helix with One Bond Missing Using Algorithm 5 · 3 and its Variant

5 · 4 · 5 · 2 Sidechain — Mainchain Hydrogen Bond Structure

For the display of sidechain — mainchain hydrogen bonds, the *diagram_side_depth[ ]* list has all its DEPTH entry fields used. The *length* field records the exact 'sequence difference' of every bond, while the *tag_depth* and *z_depth* fields record the two levels where the two extremities of the bond should be drawn. By level, I mean the position of an end_point on its appropriate residue tag. For sidechain — mainchain bonds, either the *z_depth* field or the *tag_depth* field is set to zero (e.g. no tag required) to represent the main chain connector, while the other field represents the tag length allocated to the side chain connector. Note also that in case a side chain is hydrogen bonded to more than one other atom, the lines departing from the tag are drawn in a stacked manner to avoid superposition of line drawing. Consequently, an algorithm (Algorithm 5 · 4) is devised that sets the value of both the *z_depth* and *tag_depth* field for every bond.

Before the algorithm is applied, some preprocessing is done to *diagram_side_depth[ ]* to avoid bond crossing each other. It is therefore mainly done for aesthetic reasons. Preprocessing consists of depth sorting all bonds in descending order according to their length, and then sorting them again, but this time in ascending order according to their start residue. Figure 5 · 22 illustrates some cases, applying Algorithm 5 · 4, that show why *diagram_side_depth[ ]* needs to be preprocessed in such a way. Note that in Figure 5 · 22 · a, where 2 bonds (an inter-sidechain bond and a sidechain — mainchain bond) are of equal size and have the same residue connectors, the inter-sidechain bond is drawn first. Recalling from Chapter 3 that sidechain — mainchain hydrogen bonds are listed before inter-sidechain bonds of the same 'sequence difference', the first depth sort takes care of these special cases.

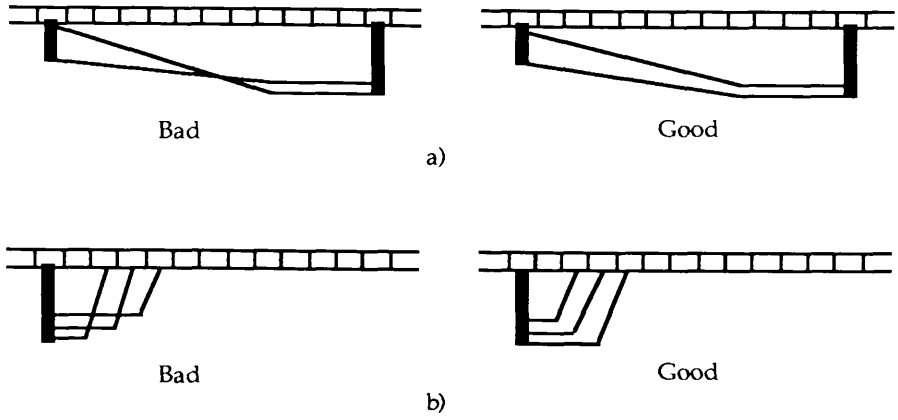


FIGURE 5 · 22 Comparison of Diagrams to Show the Need to Avoid Hydrogen Bonds Crossing Each Other

Hence, Algorithm 5 · 4 starts by processing bonds from the N-terminus to the C-terminus starting with the longest one. As for the inter-mainchain diagrams, the same *chainette[ ]* table is created. However, its purpose is different: each odd(even) element maps, this time, its corresponding left-hand (right-hand) side tag. Thus a tag is referred to by either its right-hand side or its left-hand side depending on which direction a bond departs from it. Initially, the content of *chainette[ ]* is set up in the following way: the *tag[ ]* table (already defined in the polypeptide chain structure) is scanned and for every alpha-carbon encountered (representing a residue with side chain bonding), its corresponding odd and even elements in *chainette[ ]* are allocated a value (see appendix 2) approximating the length of the side chains where connections are made.

diagram_side_depth[ ] is then processed from top to bottom. For every bond, its start and end residue are considered in order to allocate to *tag_depth* and *z_depth*, respectively, their appropriate level. For a

sidechain — mainchain bond, a question arises as to whether the start residue is the side chain connector. The answer is given by fetching the bond type from the *type* field in *diagram_line[ ]*.

In case the start residue is a side connector, then its corresponding element in *chainette[ ]* (this should be an even element as the direction of the bond is from the N-terminus to the C-terminus) is fetched and allocated to *tag_depth*. Then the content of this element is decremented by one to allow other bonds to be drawn in a stacked manner. If the start residue is a main connector, then *tag_depth* is allocated a value of zero.

The same process is performed to the *z_depth* field but through the end residue. The code for Algorithm 5 · 4 is listed in Figure 5 · 23.

5 · 4 · 5 · 3 Simultaneous Display of Two Proteins

Since the display of a second protein sequence is also allowed, so that it can be compared with the first, its tables are created the same way as the ones for the central protein under study. These tables are *diagram2[ ]*, *diagram_line2[ ]*, *diagram_main_depth2[ ]* and *diagram_side_depth2[ ]*. There is a difference in the way space memory is allocated to them. While memory allocation for tables of the central protein is static, the second protein has its tables created in a dynamic way. As a new protein is chosen for simultaneous display with the central one, a new space in core memory is allocated to all of its tables, freeing the space used by the anterior one. All the techniques used to fill out the different tables are identical to the ones used for the central model.

```
                    /*************************************************/
                    /*****************Algorithm 5 - 4*****************/
                    /*************************************************/

#define     ARRAY(length,type)      (type *) calloc(length, sizeof(type) )

diagram_side_algorithm(depth, ndepth, Tol_Num_Alpha, tag,Tol_Num_Tag )
/*****************************************************************************/
DEPTH       *depth;
int         ndepth, Tol_Num_Alpha, Tol_Num_Tag;
ATOM        *tag;
{
DEPTH       *d;
int         *x, *w, * chainette;
LINE        *l;
char        get_symbol();

chainette = ARRAY(2*Tol_Num_Alpha, int);                    /* allocate dynamically */
initialize_chainette( chainette, tag, Tol_Num_Tag) ;

for(d = depth; d - depth < ndepth; d++)  {
    l = d-> pt_line;
    switch (l->type) {
        case SIDE_SIDE_OTHER:
        case SIDE_SIDE_DISULPHIDE :
          x = chainette + ( 2 * (l-> start->amino_num ) - 1);  /* refer to right-hand side of tag */
          w= chainette  + ( 2 * (l-> connect[0] ->amino_num ) - 2 ); /* left-hand side of tag */
          d-> z_depth = (float)*x;
          d-> tag_depth = (float)*w;
          *x -= 1;
          *w -= 1;
          break ;
        case SIDE_MAIN_NH :
        case SIDE_MAIN_CO :
          x = chainette + ( 2 * (l-> start->amino_num ) - 1 );
          d-> z_depth = (float)*x;
          d-> tag_depth = 0.0;
          *x -= 1;
          break ;
        case MAIN_SIDE_NH :
        case MAIN_SIDE_CO :
          w= chainette  + (2 * (l-> connect[0] ->amino_num ) - 2 );
          d-> z_depth = 0.0;
          d-> tag_depth = (float) *w;
          *w -= 1;
          break ;
        case SIDE_MAIN_ZERONH :
        case SIDE_MAIN_ZEROCO :
        case MAIN_SIDE_ZERONH :
        case MAIN_SIDE_ZEROCO :
          d-> z_depth =          /* get approximated length of tag using appendix 2 values */
                d-> tag_depth = zero_side_level (get_symbol (l-> start->amino_type ) );
          break;
    }
}
cfree (chainette ) ;
}
```

FIGURE 5 · 23 Code for Generating Hydrogen Bond Levels in Sidechain — Mainchain Diagrams (Algorithm 5 · 4)

5 · 4 · 5 · 4 Piecewise Decomposition

In order to display fragments of the chain(s), the two (four) tables: *diagram_main_depth[ ], diagram_side_depth[ ],* (*diagram_main_depth2[ ], diagram_side_depth2[ ]*) are re-initialized by considering only the appropriate bond candidates. Linkage to either *diagram_line[ ]* (*diagram_line2[ ]*) layer 2 or layer 3 is performed appropriately through the *pt_line* pointers. The same techniques applied in section 5 · 4 · 5 · 1 and section 5 · 4 · 5 · 2 are subsequently used.

5 · 5 PHASE 3 CONSIDERATIONS

Phase 3 of the software development consists of converting residues and atoms to which the user points on the screen, or supplies as numbers through the keyboard, into pointers in the internal structure. Thereafter, appropriate system actions are taken to alter or search the corresponding internal tables.

In our model, the most important supplied command from the user is to select segments (from polypeptide chains or diagrammatic views) of the chain to be displayed with all appropriate hydrogen bonds involved in the segments. This involves removing any hydrogen bond that does not connect two residues belonging to one of the segments. This implies that one has to find the correct pointers to the different tables and at the same time make sure that these pointers are valid.

In order to ensure that no two segments are overlapping, as this would be catastrophic, a linear bit vector is created having the size of Tol_Num_Alpha. Its bit contents are initially set to zero. Once the first

extremity (e.g. start residue) of a segment is selected by the user, its corresponding bit is checked. If it is equal to 1, then this means that a segment has already been chosen in this region and the user has to make another choice. If the content of the bit is 0, it is switched to 1 and the other extremity is processed. Validation of the second extremity, and therefore of the segment, is made by checking not only its corresponding bit value (e.g. end segment bit) but all the bits between the start segment bit and the end segment bit. If it happens that one of them is set to 1, the end segment is invalid. Otherwise, it is valid and all bits between the two end points are set to 1.

5·6 CONCLUSION

The data structures shown have been mainly designed around a combination of tramline techniques, which are new strategies for indicating depth and the manipulation of three-dimensional images. A recursive algorithm (Algorithm 5·1) has been developed that differentiates between various sorts of hydrogen bond patterns and can serve as a starting point for new workers in the field. Two slightly different algorithms have been described, one for the inter-mainchain hydrogen bonds (Algorithm 5·3) and the other for the sidechain — mainchain ones (Algorithm 5·4), that allow the hydrogen bonds to be drawn so that they can be clearly seen and characteristic features portrayed. A variant to Algorithm 5·3 was also mentioned but not adopted because it is too time-consuming and, above all, the display of secondary features such as alpha-helices using this algorithm is not as informative as those with Algorithm 5·3. The coding of all routines was performed in a modular way such that it was not a too difficult task to alter or add new things as required. Finally, the manipulation of each image is fast enough to be almost complete in real time, which is some

indication that the data structures and associated algorithms are satisfactory.

Chapter 6

EXPERIMENT WITH THE SYSTEM

6·1 INTRODUCTION

The PHD system, described in the previous three chapters, is now tested by examining domain homology for a particular protein and trying to get some insights about the mystery of protein evolution. Wheat germ agglutinin (3WGA) has been chosen as the experimental example. This protein has two identical subunits, each with four 43-amino acid domains. Its three-dimensional structure was determined by X-ray crystallography at high resolution (1.8 Å) by Wright [WRIGH 87].

The next section analyses the results of the experiment, where Dr. Milner-White and myself collaborated in this protein investigation. In the final concluding section, a suggestion is made to alter the definition of secondary structure.

6 · 2 ANALYSIS OF RESULTS

6 · 2 · 1 Structure of Domains

In the present experiment, the primary concern is about the structure of domains within one subunit. Figure 6 · 1 shows a single subunit of 3WGA. The main chain is shown as a smoothed alpha-carbon plot with an averaging factor of 5. The folding of each of the four domains can be seen easily in Figure 6 · 1 from the course of the main chain, which is drawn white or with several shades of gray (intensity depth cueing effect). It is evident that the four homologous domains are arranged as a helix which is an unusual arrangement. The inter-mainchain hydrogen bonds are displayed as coloured lines joining the averaged alpha-carbon atoms. The colours represent the distance apart in sequence of the amino acids involved in each bond; yellow ones are far apart (more than five residues) and purple ones, for example, are three residues apart, as in beta-turns. Full details are given in the colour key displayed at the top right-hand corner of Figure 6 · 1. Also, from this figure, the four highly homologous domains can be detected; they are relatively lacking in secondary structure (alpha helices and beta sheets).

Figure 6 · 2 shows the diagrammatic views of the inter-mainchain hydrogen bonds where the same colouring of bonds, as in Figure 6 · 1, is used. Every hydrogen bond is clearly seen and is not obscured by atoms in front of it, as happens occasionally in Figure 6 · 1. Note that the sequence, given as the standard one-letter code (except that the N–terminus, Z, is pyrollidone carboxylate) is displayed as four superimposed segments representing the four domains. Also, note that the sequence homology between domains is 48 to 60% (without any insertions or deletions). It is rich in cystine and glycine residues and has

little secondary structures. From this figure, the relationship between bond pattern and sequence is evident. This is especially useful for representing hydrogen-bonded loop motifs, which are seen to be common in this protein. Thus, the relative lack of inter-mainchain bonds forming secondary structure appears to be compensated by those involved in loop motifs. It seems misleading to refer to regions of polypeptide that lack secondary structure as random coil since they often exhibit highly characteristic structure of their own. Figure 6 · 2 also shows that the loop motifs are highly conserved between domains. We suggest that these hydrogen-bonded motifs are conserved in much the same way that secondary structure is.

6 · 2 · 2 Features of Loop Motifs

Some aspects of the loop motifs are also investigated. In referring to amino acid sequence positions within domains, the numbering of the first domain is used to refer to homologous positions in all domains.

The yellow and orange bonds between residues 4, 9 and 10 form a G1-type beta-bulge [RICHA 81]. Since the motif exists in a loop, it may be called a beta-bulge loop [MILNE 87a]. These are frequently found motifs that can be divided into two groups, depending on the number of loop residues. The one in the agglutinin is unique because it has one more loop residue than the longest of the two common groups. However, it has the same twist usually found in such loops (see Figure 6 · 1).

The green and purple bonds between residues 19, 22 and 23 are an example of the shorter of the two groups of beta-bulge loop; they exist at the loop end of a beta-hairpin, a common situation for such motifs [MILNE 87a].

Other features are the short alpha-helix between residues 27 to 32; the so-called inverse gamma-turn [MILNE 88b] between residues 38 to 40; a classic beta-bulge [RICHA 81] involving residues 18, 36 and 37; a type II beta-turn at residues 32-35; and a type I' beta-turn between residues 13-16. The different categories of motifs can be distinguished by the hydrogen bond arrangements as well as by the dihedral angle of the main chain residues (shown in Figure 6 · 2 as red dots just above their corresponding horizontal box, when the dihedral angle is greater than zero). Note that dihedral angle displays on diagrammatic pictures have been added to the software while working on this example. All the motifs show a remarkable degree of conservation between different domains.

6 · 2 · 3 Variability of Hydrogen Bonds Involving Side Chains

Another matter under investigation is the set of bonds involving amino acid side chains. They are displayed in Figure 6 · 3. Disulphide bonds are included as if they were hydrogen bonds and are coloured buff; they are very highly conserved between domains, as they are in other homologous families [THORN 81]. Hydrogen bonds are coloured blue, red or pink, depending on the category, as explained in the key displayed at the top right hand corner of Figure 6 · 3. They are given in more precise detail in the diagrammatic views in Figure 6 · 4 (where disulphide bonds are removed to avoid cluttering in the displayed picture).

It can be seen that sidechain — mainchain bonds are more common than sidechain—sidechain ones; this is found in most proteins. All these bonds show substantial variation between the four domains and only three amino acids (see below) are involved in the same hydrogen bonding in all domains.

In all four domains, the side chain NH of the conserved GLN 36 bridges the classic beta-bulge described above by bonding to the CO group of the conserved SER 19, while the side chain CO of the same glutamine bonds to the NH of residue 5, situated in the long beta-bulge. The side chain of SER 19 bridges the short beta-bulge loop in all four domains, but by different sets of bonds in each case. Otherwise there is variation, although some homology between pairs of domains is found (e.g. GLU 72 and GLU 115). Sometimes such homology is evident even though the sequence is altered (HIS 59 and TYR 145). Sometimes the sequence is conserved, yet the side chain hydrogen bonds vary (ASN 14, ASN 15 and corresponding residues).

6·3 CONCLUSION

It is generally found that proteins are more conserved at the level of three-dimensional structure than at the sequence level [BAJAJ 84]. A structural feature that often exhibits very high homology is the disulphide bonding [THORN 81]. In contrast, electrostatic interactions are poorly conserved [BARLO 83]. However, the degree of conservation of various categories of hydrogen bond has been investigated by Dr Milner-White, Dr Poet and myself, where we compare related proteins (or domains) of known three-dimensional structure exhibiting about 50% sequence identity (article not yet published [MILNE 90]). It was found that the categories form a clear order in terms of the degree of identity found between bonds: mainchain - mainchain (secondary structure) > mainchain - mainchain (not secondary structure[1]) > bonds involving

1 The distinction between the secondary structure and non-secondary structure inter-main chain categories employs the criteria for alpha-helix and beta-sheet of Kabsh & Sander [KABSC 83].

sidechains. It was striking that inter-main chain hydrogen bonds are more conserved than bonds involving sidechains.

The results of this experiment showed that the characteristic 3WGA domain contains little regular secondary structure (alpha-helix, beta-sheet), but exhibits a set of conserved inter-mainchain hydrogen bonds. Since such 'non-secondary structure' inter-mainchain hydrogen bonds occur frequently in proteins and appear to be conserved, we suggest that the complete set of inter-mainchain hydrogen bonds be used in describing protein structure.

By contrast with the inter-mainchain hydrogen bonds, those involving side chains in the 3WGA protein are poorly conserved. Furthermore, bonds between two side chains are even less conserved. Related observations have been made for other proteins. For example, Barlow [BARLO 83] and Thornton [THORN 88] showed that ion pairs, which are mostly situated at the surface of proteins, are in general not well conserved.

The pictures in Figures 6 · 1 and 6 · 3 provide a means of portraying hydrogen bonds in the three-dimensional structure of the whole protein. Those in Figures 6 · 2 and 6 · 4 reveal the hydrogen bonding in relation to the primary structure. Generally, when pictures of whole proteins are displayed in the biological literature, they are almost invariably represented as chains with the alpha-helices and strands of beta-sheet singled out in some way (see Chapter 2). Although this is a convenient representation, it does make the assumption that these particular features are the most important ones. Hydrogen bonds between main chain atoms are well-known to be the basis for alpha-helices and beta-sheet. However, many other inter-mainchain hydrogen bonds occur in proteins. Many of them appear to stabilize loop motifs. Although non-repetitive,

unlike alpha-helices and beta-sheet, loop motifs are on the whole well conserved in evolution. This experiment demonstrates the usefulness of my displays. They provide representations of whole proteins that display all inter-mainchain hydrogen bonds and not just those in repetitive secondary structure elements.

An advantage of diagrammatic pictures, such as the ones in Figure 6 · 2 and Figure 6 · 4, is that every single hydrogen bond is clearly seen since it cannot be obscured by atoms in front of it, as happens occasionally in the displays of Figure 6 · 1 and Figure 6 · 3. These schematic pictures, especially ones like that in Figure 6 · 2, are useful for studies aimed at predicting the three-dimensional structure of proteins from sequence, because both secondary structure and conserved loop motifs can be easily visualized in the context of the sequence. Although the diagrammatic pictures are two-dimensional represention, the bond patterns, which exist as three-dimensional features of proteins add an extra depth to the pictures. This clue makes the schematic pictures look as they are viewed in two and a half dimensions.

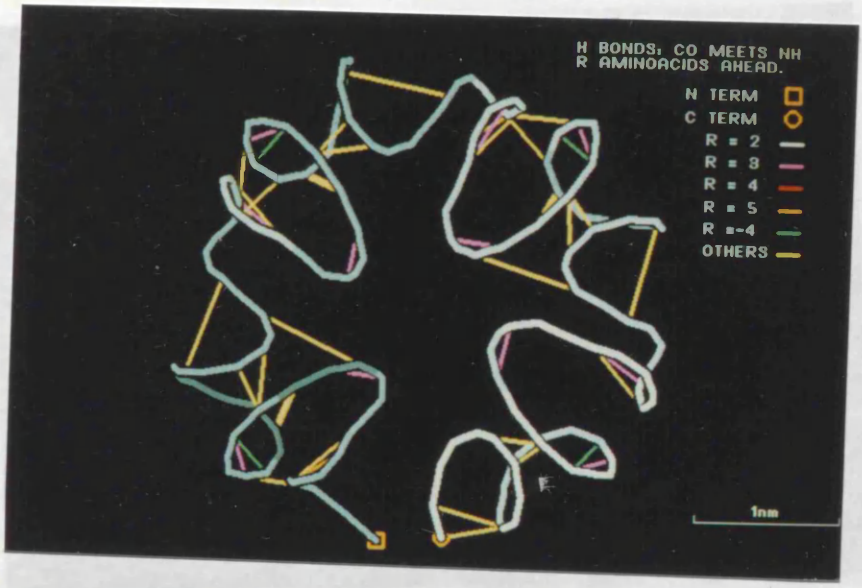


FIGURE 6 · 1 The Inter-Mainchain Hydrogen Bonds in the Three-Dimensional Structure of Wheat Germ Agglutinin (3WGA)

This and subsequent pictures in this Chapter are computed from the X-ray crystallographic structure of the 3WGA protein [WRIGH 87]. The main chain is represented as a smoothed alpha-carbon plot. Inter-mainchain hydrogen bonds are portrayed as coloured lines joining the positions of the relevant smoothed alpha-carbon atoms. Where both sets of CO and NH groups of pairs of amino acids are bonded, a thicker single line is drawn. The colour represents the distance apart in the sequence of two amino acids joined by a bond; in the colour key shown at the top right-hand corner, an integer R defines this distance as: the CO group of residue i binds to the NH group of residue i + R.

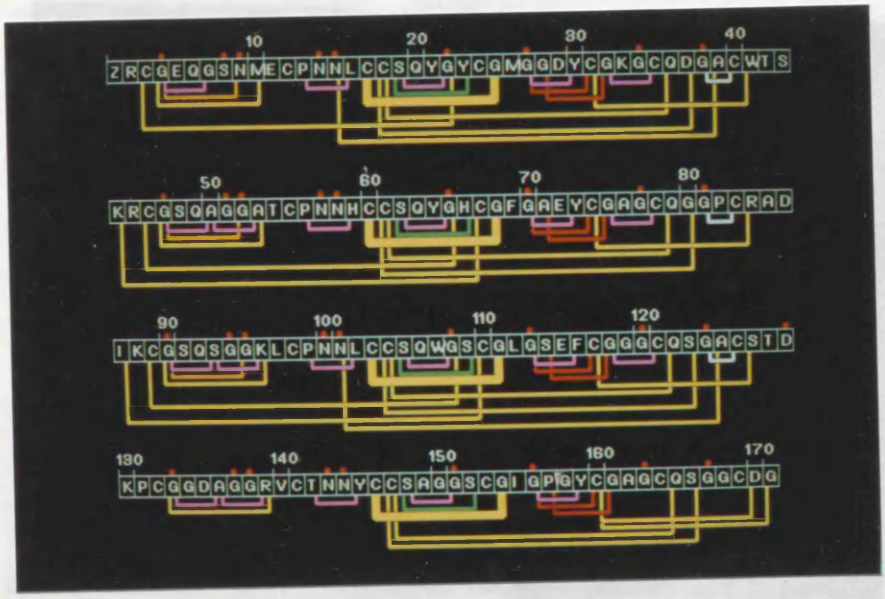


FIGURE 6 · 2 Diagram of the Inter-Mainchain Hydrogen Bonds of the Four Domains of Wheat Germ Agglutinin (3WGA)

The sequence, given as the standard one-letter code (except that the N-terminus, Z, is pyrollidone carboxylate), is divided into four domains. Hydrogen bonds are drawn in colour in a pipe-like representation. As before, where NH and CO groups of a pair of amino acids are bonded, a single thick line is drawn. The letter for each amino acid is surrounded by a small box. The left-hand side of the box corresponds to the NH part and the right-hand side to the CO part. Hydrogen bonds are drawn at one side or the other of the box to represent NH or CO groups, except for the paired bonds mentioned above, which are drawn in the middle. A few atoms, for example the oxygen of GLY 27, bond to more than one other atom. Whenever dihedral angles are greater than zero, a red dot is displayed above their corresponding 'box'.

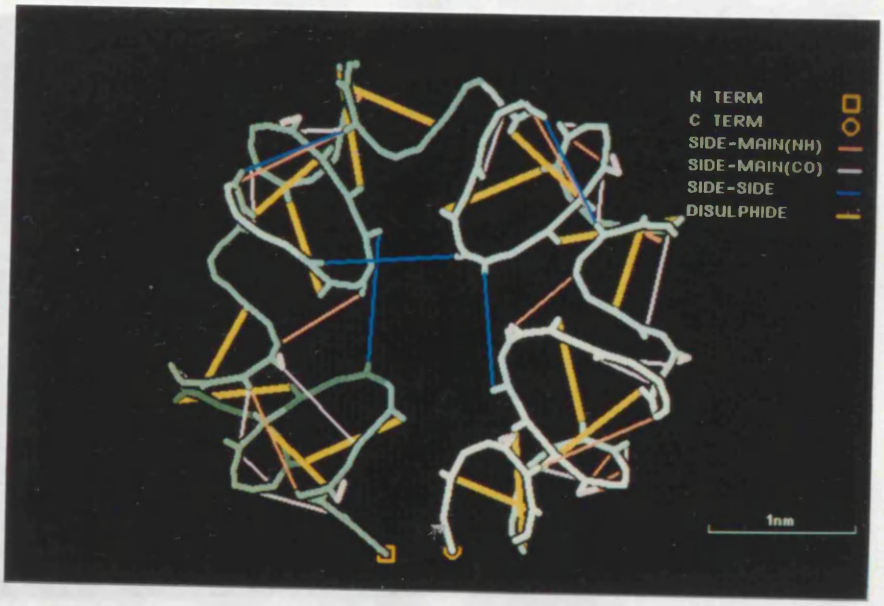


FIGURE 6 · 3 Hydrogen Bonds Involving Side Chains and Disulphide Bonds in the Three-Dimensional Structure of Wheat Germ Agglutinin (3WGA)

The main chain is represented as in Figure 6·1 and side chains involved in bonding are represented by tags which project from it. Their direction is computed by bisecting the obtuse angle formed by the smoothed alpha-carbon coordinates of the amino acid in question and those of its two neighbours. Tags are all the same length but may appear foreshortened or obscured. Disulphide bonds are drawn as buff-coloured lines between the ends of tags. Hydrogen bonds are portrayed as lines of other colours, either between the ends of tags (for inter-sidechain bonds, blue), or between one tag and one smoothed alpha-carbon (for sidechain — mainchain bonds, red or pink). Hydrogen bonds where the side chain bonds to a main chain atom of the same residue are represented as thick oblique flag-like features using the same colour-coding as for the other bonds.

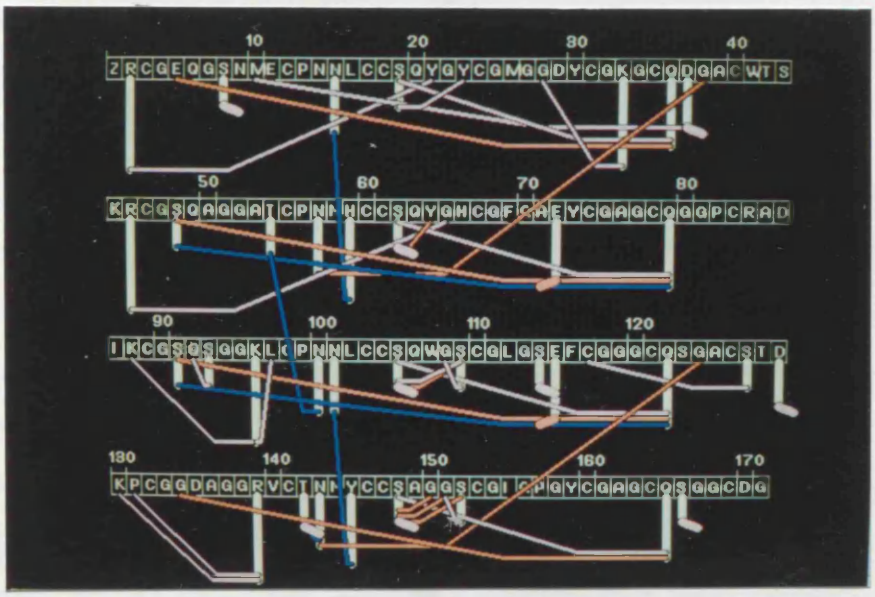


FIGURE 6 · 4 Diagram of the Hydrogen Bonds Involving Side Chains in the Four Domains of Wheat Germ Agglutinin (3WGA)

The sequence is divided into the four domains. The letters in boxes represent amino acids as in Figure 6 · 2, and correspond to the main chain part of residues. Side chains involved in bonding are represented by white vertical lines (tags), their length approximating to the length of the side chain. Hydrogen bonds are portrayed as coloured lines, either between the ends of tags (for inter-sidechain bonds, blue), or between one tag and one main chain 'box' (for sidechain — mainchain bonds, red or pink). Hydrogen bonds between side chain and main chain atoms from the same residue are portrayed as thick oblique flag-like features (one is seen at ASP 129).

Chapter 7

CONCLUSIONS

7·1 GENERAL CONCLUSIONS AND CONTRIBUTIONS

This research has described the development of new ways of picturing proteins that assist in understanding both their three-dimensional structure and the relationships between it and their sequence.

The PHD system has provided a working environment for the biochemist allowing him, or her, to manipulate models of known protein three–dimensional structure. Pictures of the polypeptide chain with either inter-main chain hydrogen bonds, or those involving side chain — main chain bonds, can be displayed. Interesting sections of proteins can be selected, showing chain structure, atomic details or diagrammatic views. Colour coding has been used to differentiate between the different types of hydrogen bond found. A recursive

algorithm for the inter-mainchain hydrogen bond pictures has been developed that differentiates between various sorts of hydrogen bond patterns. This algorithm was particularly useful for distinguishing between hydrogen bonds belonging to a parallel beta sheet or an anti-parallel one, which was not easily differentiated in the early pictures of the project [POET 86].

A further novel computer-generated graphical (called diagrammatic view) display technique for proteins was also described, in which the backbone is represented as a horizontal line and the inter-mainchain hydrogen bond structure (or the sidechain — mainchain one) is related to the sequence. For such views, two slightly different algorithms have been developed, one for the inter-mainchain hydrogen bonds and the other for those involving sidechains, that allow every hydrogen bond to be clearly seen since it cannot be obscured by atoms in front of it, as happens occasionally in the displays described above. Another advantage of this type of display is that all secondary structure features and particularly short loop motifs can be readily portrayed. Furthermore, the relationship between bond pattern (and consequently the three-dimensional structure) and the sequence is evident. Although, the diagrammatic pictures are two-dimensional represention, the bond patterns which are three-dimensional features of proteins, add an extra depth to the pictures. This clue makes the schematic pictures look as they are viewed in two-dimension and a half. No other known computer-generated pictures provide such comprehensive information about this aspect of proteins which is a major requirement for predicting protein structure from primary structure.

Another main point concerns the way whole proteins are viewed. To be appreciated in three-dimensions they have to be simplified. From currently available systems describing methods to visualize secondary

structures (see Chapter 2), it is usually impossible for a user to see through the object displayed. Moreover, an investigator is not in a state to know how many hydrogen bonds are involved in secondary structure features. Furthermore, there is no way a researcher can investigate unpredictable features such as loop motifs from these models, where all that can be seen are some known specific features. A major factor that should be considered is that it is important not to build in preconceived ideas of what we expect to see, since this will make it harder to discover new features. On the other hand, the well known structures should be clearly visible. This balancing act is another desirable feature that a system should achieve. The currently available methods select certain hydrogen-bonded features in an arbitrary way, and it is better, instead, to depict all inter-mainchain hydrogen bonds. The pictures of the polypeptide chain with the inter-mainchain hydrogen bonds and their corresponding diagrammatic views provided a means to do it.

The strategy I adopted in picturing proteins is based on the idea of multiple simplified displays, where only important information is displayed, so that the investigator is not swamped by unnecessary detail. These pictures can be manipulated interactively and displayed simultaneously, taking advantage of window management techniques incorporated in the CG1.

The question of how to represent depth is also of special concern in the present study. Ways of visualizing three-dimensional objects on a two-dimensional screen without disturbing the concentration of the viewer has been one of the goals of my research. A special feature of my system is the way I have developed new strategies for indicating depth. Some new techniques such as tramlines and the use of circular brushes have been developed which provide an effective three-dimensional illusion for the user. This combination of techniques has made it

possible to create a model image of a protein which stimulates the imagination of biochemists, while minimizing time delays for display or manipulation of the images. These methods can be applied generally over various fields of Computer Graphics such as CAD/CAM systems.

Several protein displays have been illustrated. An analysis using the system has been carried out in Chapter 6 using the protein wheat germ agglutinin. The results showed that, for regions of proteins that lack secondary structure, the inter-mainchain hydrogen bonds of loop motifs appear to be highly conserved. It is suggested that the use of the term secondary structure is at times inappropriate, since it is often used to describe the most highly conserved hydrogen-bonded regions of protein structure, and that it would be better to include loop motifs in this category.

This research has provided a new tool to help prediction of three–dimensional structure from sequence information. Its special importance is that it allows the relationship between three-dimensional structure and sequence to be readily investigated. This work has also provided a substantial increase in understanding of protein structural features, knowledge about which will assist in prediction work.

7·2 FUTURE DIRECTIONS

While the current application is based on a CG1 colour workstation, a more portable window system, such as the X Window Sytem (version 11), is desirable. In such an environment, writing an application program makes it portable to any hardware supporting an X server by

simply recompiling it; no changes in the code will be required ([SCHEI 86, SCHEI 88]).

Visualization of proteins have focused essentially on two levels :the folding of polypeptide chains and the relationship between three-dimensional structures and their sequence. Further levels could be added and constitute a further extension to the project. These include the development of a means of investigation of surfaces and interfaces, as well as of the overall shape of subunits.

It is likely that the ability to recognize loop motifs will become crucial in the prediction of protein structure as a whole. The problem, at present, with predicting loop motifs is not that they do not exhibit marked propensities for particular amino acids at positions along a given motif, but that there is not enough data in the PDB. There also exist a number of other different sorts of motifs. This suggests that there is scope for further research because it is only when any structural group has been clearly differentiated that prediction methods can be optimized.

Since hydrogen bonds are key components of both secondary structure and loop motifs, the diagrammatic views are likely to be helpful, both in identifying such features and in establishing their sequence propensities. Future directions will be to pursue this idea and use it as a basis for a novel prediction method which will be employed for sequence analysis of proteins, and protein families, whose structure one wishes to predict. In the first stage, the sets of amino acid propensities for individual loop motifs could be used to scan families of sequences of known three-dimensional structure to examine how useful are the predictions that can be made. It will then be possible at a later stage to correlate, for example, rapidly a three-dimensional motif with the set of

amino acid sequences at that position in a family of proteins.

Extensive capabilities on the diagrammatic views could be added within the existing framework. Although color coding is mainly used to differentiate between different types of hydrogen bond found, it assumes a familiarity with the code. There should be an automatic facility that detects all features of protein structure, especially loop motifs. A novice user can be acquainted with these representations by requesting, for example, at a first stage that only alpha-helices be displayed. At a next stage, he may ask for a specific loop motif to be displayed. This facility would also be helpful for more frequent users. As more loop motifs are discovered in the biological literature, the ability of the user to memorize them becomes more and more of a chore; this facility will help solve the problem.

Appendix 1

AMINO ACID SYMBOLS

Amino Acid	Three-letter symbol	One-letter symbol[1]
Alanine	ALA	A
Arginine	ARG	R
Asparagine	ASN	N
Aspartic acid	ASP	D
Cysteine	CYS	C
Glutamine	GLN	Q
Glutamic acid	GLU	E
Glycine	GLY	G
Histidine	HIS	H
Isoleucine	ILE	I
Leucine	LEU	L
Lysine	LYS	K
Methionine	MET	M
Phenylalanine	PHE	F
Proline	PRO	P
Serine	SER	S
Threonine	THR	T
Tryptophan	TRP	W
Tyrosine	TYR	Y
Valine	VAL	V

1 Amino acids are ordinarily designated by three-letter symbols, but a set of one letter symbols has also been adopted to facilitate comparative display of amino acid sequences of homologous proteins.

Appendix 2

APPROXIMATION OF SIDE CHAIN LENGTHS

The length of each 'tag' approximates the length of the corresponding side chain.

Amino Acid	One-letter symbol	Tag length
Alanine	A	0
Arginine	R	14
Asparagine	N	8
Aspartic acid	D	7
Cysteine	C	6[1]
Glutamine	Q	10
Glutamic acid	E	9
Glycine	G	0
Histidine	H	12
Isoleucine	I	0
Leucine	L	0
Lysine	K	13
Methionine	M	0
Phenylalanine	F	0
Proline	P	0
Serine	S	4
Threonine	T	5
Tryptophan	W	0
Tyrosine	Y	11
Valine	V	0

1 While Cysteine has potential side chain atom bonding, its tag is not displayed in diagrammatic views as it would confuse the picture.

Appendix 3

GRAPHICS ROUTINES

This appendix describes all the graphics library routines used in the PHD system. Their code can be found in the file *window.c*. These procedures are designed to allow simplified access from C language programs to the Angel graphics package supplied with the CG1 workstation. All of them were implemented by a collaboration between Dr. Poet and myself.

NAME

WINDOW

SYNOPSIS

win_open(obj_x1, obj_x2, obj_y1, obj_y2, ndc_x1, ndc_x2, ndc_ 1, ndc_y2, title)

```
double  obj_x1,obj_x2,obj_y1,obj_y2;
double  ndc_x1, ndc_x2,ndc_y1,ndc_y2;
char    *title;
```

win_close(winnum)

```
int     winnum;
```

win_show(winnum)

```
int     winnum;
```

win_wipe(winnum,colour)

```
int     winnum;
int     colour;
```

win_colour(winnum, col_file)

```
int     winnum;
char    *col_file;
```

win_enable(winnum,event_fun)

```
int     winnum;
int     (*event_fun)();
```

win_disable(winnum)

```
int     winnum;
```

PARAMETERS

winnum — The integer window identifier, as returned by *win_open*

obj_x1 etc... — The world coordinate system for the window

ndc_x1 etc... — The screen position of the window on normalized device coordinates

title — The window title

colour — The whole window is set to this colour

col_file — A file containing colour information (see COLOUR section)

```
event_fun(winnum,device,x,y)
    int     winnum;
    char    *device;
    double  x, y;
```

A function to be executed when a mouse interrupt is generated.

device — either "mouse1" or "mouse2", depending on the number of clicks.

x, y — World coordinates of the mouse when clicked.

DESCRIPTION

win_open creates a new window, while *win_close* destroys the window. *win_show* updates all the changes made to the window.

win_wipe clears the window, setting it to the desired colour.

win_colour processes a colour definition file (see COLOUR section for its implementation).

win_enable provides an interrupt handling function, which is executed when the mouse is clicked in a enabled window. The 4 parameters specify the window id, the number of clicks (as a string) and the x, y coordinates of the mouse when clicked.

NAME

BRUSH

SYNOPSIS

brush_thick(winnum, thick)

```
int     winnum, thick;
```

brush_colour(winnum, colour)

```
int     winnum, colour;
```

brush_dash(winnum, obj_dash)

```
int     winnum;
double  obj_dash;
```

brush_xy(winnum, obj_x, obj_y)

```
int     winnum;
double  obj_x,obj_y;
```

PARAMETERS

winnum	The window id
thick	The line thickness in pixels
colour	The colour id
dash	The length of dashed lines in world coordinates (≤ 0.0 => solid lines)
obj_x,obj_y	x,y coordinates in world coordinates

DESCRIPTION

The brush thickness, colour, dash length and position is changed by these routines.

NAME

DRAW

SYNOPSYS

draw_dot(winnum)

```
int     winnum;
```

draw_line(winnum, obj_x, obj_y)

```
int     winnum;
double  obj_x, obj_y;
```

draw_circle(winnum, obj_r)

```
int     winnum;
double  obj_r;
```

draw_box(winnum, obj_r)

```
int     winnum;
double  obj_r;
```

draw_cross(winnum, obj_r)

```
int     winnum;
double  obj_r;
```

PARAMETERS

winnum	Window id
obj_x, obj_y	coordinates of end of line
obj_r	Radius of circle, box or cross marker

DESCRIPTION

Line is drawn from the current brush position, updating it; dot, circle, box and cross are drawn at the current brush position.

NAME

TEXT

SYNOPSIS

pen_colour(winnum, colour)

```
int     winnum, colour;
```

pen_xy(winnum, obj_x, obj_y)

```
int     winnum;
double  obj_x, obj_y;
```

write_str(winnum, text)

```
int     winnum;
char    *text;
```

write_ch(winnum, c)

```
int     winnum;
char    c;
```

read_line(win, buf, max_buf)

```
int     win, max_buf;
char    buf[ ];
```

PARAMETERS

winnum	Window id
obj_x, obj_y	World coordinates for pen position, initially at top left and corner
text	Null terminated array of characters
buf	Array of characters
max_buf	Array bound
c	Character to be output to screen

DESCRIPTION

No filtering of special output characters is performed. The *read_line* function reads up to the next newline character.

NAME

MENU

SYNOPSIS

menu_open(winnum, title, [item_name, action,]....,0)

```
int       winnum;
char      title [ ], item_name [ ];
INT_FUN   action;
```

menu_show(menu, obj_x, obj_y)

```
int       menu;
double    obj_x, obj_y;
```

menu_hide(menu)

```
int       menu;
```

menu_mark(menu, item)

```
int       menu, item;
```

PARAMETERS

winnum — Window id

title — The menu title

item_name — The name of the menu item

action(menu, item)

int menu, item;

The function to be called when that menu item is chosen

menu — The menu id, generated by menu_open

obj_x, obj_y — The top left hand corner of the menu, relative to its window

item — item number (starting from 0) in the menu

DESCRIPTION

Menus are associated with windows, so that *menu_open* must be provided with a window id as well as the title. This is followed by a zero terminated list of item name/action pairs for each item in the

menu. When a menu item is selected, the appropriate action function is called, and is passed two integer parameters, the menu id and the item number. Items are numbered starting from 0.

NAME

COLOUR

SYNOPSIS

makecol **colour_file**

DESCRIPTION

The *makecol* program takes a sequence of colour definition files and generate the appropriate .h files for inclusion in C programs. This allows colours to be refered to by name in the program.

The format of a colour definition file, as used by the *win_colour* function and the *makecol* program, is as follows:

Line 1: A header line which is ignored.
Lines that follow are in one of the two following forms:

colour_name	*1*	red_value	green_value	blue_value
		or		
colour_name	*n*	red_range	green_range	blue_range

The *colour_name* is just an identifier for that particular colour, to be used in the program. The *red, green and blue values* are real numbers between 0.0 and 1.0, denoting the intensities of these primary colours. The number *n* which appears in the second case is the number of shades of the colour required. The *range* of colours is in the form 0.25-0.75, denoting colour intensities between 0.25 (first shade) and 0.75 (last shade).

The implementation of the *makecol* program is given next, together with an example that illustrates the format of a colour definition file (input) and its appropriate .h file (output). Next follows the code of the *win_open* function (in order to set-up a colour/intensity lookup table for any coloured windows created, using a CG1 workstation).

```
                        /************************/
                       /****** makecol.c  file*****/
                        /************************/


#include <stdio.h>
#include <sys/types.h>
#include <panels.h>

#define MAX_COLOURS    256

main(argc, argv)
/**************/
        int     argc;
        char    *argv[];
        {
        FILE    *colc, *colh;
        char    name[40];
        int     range, i, c;
        double  r, r2, rinc, g, g2, ginc, b, b2, binc;

        sprintf(name, "%s.h", argv[1]);
        colc = fopen(argv[1], "r");
        colh = fopen(name, "w");
        if (!colh)
                die("Cannot open colour files");        / * function that kills the process */

        while (getc(colc) != '\n')                      /*flush first line  */
                ;

        c = 2;
        while (fscanf(colc, "%s%d", name, &range) != EOF)
                {
                if (range == 1)
                        {
                        fscanf(colc, "%lf%lf%lf", &r, &g, &b);
                        fprintf(colh, "define %s\t(%d)\n",name, c);
                        }
                else
                        {
                        fscanf(colc,  "%lf%lf%lf%lf%lf%lf", &r, &r2, &g, &g2, &b, &b2);
                        rinc = (-r2 - r) / (range - 1);
                        rinc = (-g2 - g) / (range - 1);
                        rinc = (-b2 - b) / (range - 1);
                        fprintf(colh, "#define %s (x) \t (%d + (int) (%d* (x))) \n",
                                name, c, range - 1);
                        }
                c += range;
                }
        }
```

INPUT: *colours* file

NAME	#VALUE	RED	GREEN	BLUE
WHITE	1	1.0	1.0	1.0
BLACK	1	0.0	0.0	0.0
BLUE	1	0.0	0.0	1.0
GREEN	1	0.0	1.0	0.0
PINK	1	0.73	0.55	0.55
RED	1	1.0	0.0	0.0
YELLOW	1	1.0	1.0	0.0
CHAIN	8	1.0-0.2	1.0-0.2	1.0-0.2
NITROGEN	3	0.0-0.0	1.0-0.2	0.0-0.0
SIDECARBON	3	1.0-0.2	0.0-0.0	0.0-0.0
OXYGEN	3	0.0-0.0	0.0-0.0	1.0-0.2
MAINCARBON	3	1.0-0.2	1.0-0.2	1.0-0.2

makecol
program takes the input definition file
colours
and generates the
colours.h
for inclusion in C programs

OUTPUT: *colours.h* file

```
#define  WHITE              (2)
#define  BLACK              (3)
#define  BLUE               (4)
#define  GREEN              (5)
#define  PINK               (6)
#define  RED                (7)
#define  YELLOW             (8)
#define  CHAIN              (9   + (int) (7 * (x)))
#define  NITROGEN(x)        (17 + (int) (2  * (x)))
#define  SIDECARBON(x)      (20 + (int) (2 *  (x)))
#define  OXYGEN(x)          (23 + (int) (2 *  (x)))
#define  MAINCARBON(x)      (26 + (int) (2 *  (x)))
```

```
#include <sys/types.h>          /* library calls */
#include <panels.h>
#include <menus.h>
#include <panels.h>
#include <win.h>
#include <brush.h>
#include <pen.h>

#define MAX_Win          20
#define MAX_BRUSH_SIZE 5
#define MAX_COLOURS    256

typedef struct      {                           /* data structure of any window */
    Window          *win;
    Raster          *ras;
    short           panel;
    Brush           *brush[MAX_BRUSH_SIZE];
    Brush           *b;             /* current brush */
    int             thick;          /* current thickness */
    int             colour;         /*current colour */
    int             dash;           /*current dash length (0 => solid)  */
    int             x, y;           /* current brush position */
    Pen             *pen;
    MenuFrame       *mf;
    double          x_scale, y_scale, x_shift, y_shift;
    int             COLOUR;         /* > 0 if yes */
    int             (*dev_fun)( );  /* zero if mouse not enabled */
    } Win;

    Win             winlist[MAX_Win];
    Colour          coltab[MAX_COLOURS];
    int             ncolours = 0;


        /*****************************************************************/
                        win_colour(winnum, col_file)
        /*****************************************************************/
    int     winnum;
    char    *col_file;
    {
    FILE    *colc;
    char    name[40];
    int     range, i, c;
    double  r, r2, rinc, g, g2, ginc, b, b2, binc;

            /*************************************************/
            /**********check if on a colour machine************/
            /*************************************************/

    if (PaletteDepth() == 1)                    /*  don't implement lookup table if
            return;                             /*  monochrome system */

    colc = fopen(col_file, "r");
    if (!colc)
            die("Cannot open colour files");
```

```
while (getc(colc) != '\n')                    /*  flush first line  */
        ;

rinc = ginc = binc = 0.0;                     /*  make sure set if range == 1  */

        /*********************************************************/
        /**************LOOKUP  TABLE  SET_UP*****************/
        /*  The coltab array contains ncolours entries, describing       */
        /*  the colours required by the application. Each colour is      */
        /*   specified by giving the intensities of red, green and       */
        /*   blue that make up the colour; each intensity is an integer  */
        /*   between 0 and COLOUR_MAXI inclusive. (See [KHAN 86] */
        /*      for  more  details)*************************************/
        /*********************************************************/
coltab[0].red     = 0;
coltab[0].green   = 0;
coltab[0].blue    = 0;
coltab[0].type    = COLOUR_EXACT;          /* use programmable colour */
coltab[1].red     = COLOUR_MAXI;           /*  maximum colour value   */
coltab[1].green   = COLOUR_MAXI;
coltab[1].blue    = COLOUR_MAXI;
coltab[1].type    = COLOUR_EXACT;
c = 2;
while (fscanf(colc, "%s%d", name, &range) != EOF)
        {
        if (range == 1)
                {
                fscanf(colc, "%lf%lf%lf", &r, &g, &b);
                }
        else
                {
                fscanf(colc, "%lf%lf%lf%lf%lf%lf", &r, &r2, &g, &g2, &b, &b2);
                rinc = (-r2 - r) / (range - 1);
                ginc = (-g2 - g) / (range - 1);
                binc = (-b2 - b) / (range - 1);
                }
        for (i = 1; i <= range; i++,  c++)
                {
                if (c >= MAX_COLOURS)
                        die("Too many colours");
                coltab[c].red     = (int)  (r * COLOUR_MAXF);
                coltab[c].green   = (int) (g * COLOUR_MAXF);
                coltab[c].blue    = (int) (b * COLOUR_MAXF);
                coltab[c].type    = COLOUR_EXACT;
                r+= rinc;
                g += ginc;
                b += binc;
                }
        }
ncolours = c;

if (PaletteRequest(winlist[winnum].panel,
                coltab, ncolours) != ncolours) ) /*used to request a number of      */
        die("Cannot assign enough colours");  /* ...colours from the palette      */
winlist[winnum].COLOUR = 1;
}
```

References

[AGOST 79] AGOSTON, G.A. *"Color Theory and its Application in Art and Design"* (edition by MacAdam,A.L.) Vol.**19** Springer–Verlang Berlin Heidelberg (1979)

[ALLEN 82] ALLEN, R.B. *"Cognitive Factors in Human Interaction with Computers"* in Badre and Shneiderman, Directions in Human/Computer Interaction, Ablex Publishing Corp., Norwood, N.J. (1982)

[ANFIN 73] ANFINSEN, C.B. *"Principle that Govern the Folding of Protein Chains"* Science Vol. **181** pp.223-230 (1973)

[ARTWI 84] ARTWICK, B.A. *"Applied Concepts in Microcomputer Graphics"* Prentice-Hall, Inc., Englewood Cliffs, N.J. (1984)

[BAJAJ 84] BAJAJ, O. & BLUNDELL, T. *"3-D Structure Conservation in Proteins"* Ann. Rev. Bioeng. Vol.**13** pp.453-492 (1984)

[BAKER 80] BAKER, E.N. & DODSON, E.J. *"Crystallographic Refinement of the Structure of Actinidin at 1.7 Å Resolution by Fast Fourier Least-Squares Methods"* Acta Crystallogr. Sect.A Vol.**36** pp. 559-572 (1980)

[BAKER 84] BAKER, E.N. & HUBBARD, R.E. *"Hydrogen Bonding in Globular Proteins"* Prog.Biophys. Molec. Biol. Vol.**44** pp. 97-179 (1984)

[BARBE 83] BARBER, R.E. & LUCAS, H.C. *"System Response Time, Operator Productivity, and Job Satisfaction"* CACM, Vol.**26** Num.11 pp972-986 (1983)

[BARLO 83] BARLOW, D.J. & THORNTON, J.M. *"Ion-Pairs in Proteins"* Journal of Molecular Biology Vol.**168** pp.867-885 (1983)

[BARRY 69] BARRY, C.D., ELLIS, R.A., GRAESSER,S. & MARSHALL,G.R. *"Pertinent Concepts in Computer Graphics"* (edit. by Faiman, M. and Nieuergelt, J.) University of Illinois Press, Urbana pg.104 (1969)

[BASH 83] BASH, P.A., PATTABIRAMAN, N., HUANG, C. FERRIN, T.E. & LANGRIDGE, R. *"Van-der-Waals Surfaces in Molecular Modelling: Implementation with Real_Time Computer Graphics"* Science Vol.**222** pp. 1325-1327 (1983)

[BURRI 84] BURRIDGE, J.M. & TODD, S.J.P. *"An interactive Interface for Protein Secondary Structural Representation"* Journal of Molecular Graphics Vol.**2** Num.2 pg. 53 (1984)

[BURRI 89] BURRIDGE, J.M. et al *"The WINSOM solid Modeller and its Application to Data Visualization"* IBM Ssystem Journal Vol.**28** Num.4 pp. 548-568 (1989)

[BERNS 77] BERNSTEIN, F.C. KOETZLE, T.F. WILLIAMS, G.J.B. MEYER, E.F. BRICE, M.D. ROGERS, J.R. KENNARD, O. SHIMANOUCHI, T. & TASUMI, M. *"The Protein Data Bank: A Computer Archive"* Journal of Molecular Biology. Vol.**112** pp. 535-542 (1977)

[CLARK 88] CLARKE, B. *"A Molecular Graphics Suite of Programs for a Microcomputer to Display Molecules from Cambridge Crystallographic Data Files and the Alpha-Carbon Backbone of Proteins from Protein Data Bank Crystal Files"* Computer Chemistry. Vol.**12** Num.1 pp.65-82 (1988)

[CONNO 81] CONNOLLY, M.L. *"Protein Surfaces and Interiors"* Ph.D. Thesis, University of California, Berkeley (1981)

[CONNO83a] CONNOLLY, M.L. *"Solvent-Accessible Surface of Protein and Nucleic Acids"* Science Vol.**221** pp.709-713 (1983)

[CONNO 83b] CONNOLLY, M.L. *"Analytical Molecular Surface Calculation"* Journal of Applied Crystallography Vol.**16** pp.548-558 (1983)

[COOK 80] COOK, C.R. *"Best Sorting Algorithm for Nearly Sorted Lists"* Communications of the A.C.M. Vol.**23** Num.11 pp. 620-624 (1980)

[CURRE 86] Current Communications in Molecular Biology *"Computer Graphics and Modecular Modeling"* Cold Spring Harbor Laboratory (1986)

[FELDM 73] FELDMAN, R.J. & BACON, C.R.T. *"Versatile Interactive Graphics Display System for Molecular Modelling by Computer"* Nature Vol.**244** pp.113-115 (1973)

[FELDM 76] FELDMANN, R.J. *"ANSOM - Atlas of Molecular Structure on Microfiche "* Tracor Jitco Inc., MO, USA (1976)

[FELDM 80] FELDMAN, R.J. *"Molecular Dynamics of Bovine Pancreatic Trypsin Inhibitor"* Colour film (1980)

[FERRI 80] FERRIN, T.E. & LANGRIDGE, R. *"Interactive Computer Graphics with the UNIX Time-Sharing System"* Computer Graphics Vol.**13** pp.321-330 (1980)

[FERRI 84] FERRIN, T.E., CONRAD, C.H., LAURIE, E.J. & LANGRIDGE, R. *"Molecular Interactive Display and Simulation: MIDAS"* Journal of Molecular Graphics. Vol.**2** Num.2 pg.55 (1984)

[FERRI 88a] FERRIN, T.E., CONRAD, C.H., LAURIE, E.J. & LANGRIDGE, R.*"The MIDAS Database System"* Journal of Molecular Graphics Vol.**6** Num.1 pp.2-12 (1988)

[FERRI 88b] FERRIN, T.E., CONRAD, C.H., LAURIE, E.J. & LANGRIDGE, R. *"The MIDAS Display System"* Journal of Molecular Graphics Vol.**6** Num.1 pp.13-27 (1988)

[FILMA 82] FILMAN, D.J., BOLIN, J.T., MATTHEWS, D.A. & KRAUT, J. *"Crystal Structures of Eschericchia and Lactobacillus Dihydrofolate Reductase Refined at 1.7 Angstroms Resolution"* Journal of Biol. Chem. Vol.**257** pp.13663-13672 (1982)

[FOLEY 82] FOLEY, J.D. & VAN DAM,A. *"Fundamentals of Interactive Computer Graphics"* Addison Wesley Publishing Company (1982)

[GENIX 84] GENIX Programmer's Manual. Vol.**1**, Whitechapel Computer Works Ltd.,London (1984)

[GEORG 85] GEORGE, D.C. *"Protein Identification Resource"* PIR Report REL-0185 - National Biomedical Research Foundation - Georgetown University Medical Center, Washington D.C. (1985)

[HEARN 86] HEARN, D. & BAKER, M.P. *"Computer Graphics"* Prentice-Hall International Editions (1986)

[HODGS 85] HODGSON, G.M. & RUTH, S.R. *"The Use of Menus in the Design of On-Line Systems: A Retrospective View"* ACM/SIGCHI Vol.**17** Num.1 pp-16-22 (1985)

[HOLME 82] HOLMES, M.A. & MATTHEWS, B.W." *Structure of Thermolysin Refined at 1.6 Angstroms Resolution"* Journal of Molecular Biology Vol.**160** pp.623-646 (1982)

[HOPGO 86] HOPGOOD, F.R.A, DUCE, D.A, FIELDING, E.V.C., ROBINSON, K. & WILLIAMS, A.S. *"Methodology of Window Management"* Eurographics Seminars Springer-Verlag (1986)

[HOPGO 83] HOPGOOD, F.R.A, DUCE, D.A, GALLOP, J.R. & SUTCLIFFE, D.C. *"Introduction to the Graphical Kernel System G.K.S."* A.P.I.C. Studies in Data Processing Num.19 Academic Press pp.67-81 (1983)

[IGA 84] IGA, Y. & YASUOKA, N. *"VENUS - A Program to Display Protein Structure using Raster Colour Graphics"* Journal of Molecular Graphics Vol.**2** Num.3 pp.79-82 (1984)

[KABSC 83] KABSCH, W. & SANDER, C. *"Dictionary of Protein Secondary Structure: Pattern Recognition of Hydrogen-Bonded and Geometrical Features"* Biopolymers Vol.**22** Num.2 pp.2577-2637 (1983)

[KAMPH 84] KAMPHUIS, K.H., KALK, K.H., SWARTE, M.B.A. & DRENTH, J. *"Structure of Papain Refined at 1.65 Å Resolution"* Journal of Molecular Biology Vol.**179** pp.233-256 (1984)

[KENDR 61] KENDREW, J.C. *"The Three-dimensional Structure of a Protein Molecule"* Scientific American pp.96-110 (Dec.1961)

[KHAN 86] KHAN, N. & PODOLSKI, Z. *"A Programmer's Guide to Graphics on Whitechapel Workstations"* Department Report CSC/86/D1 Computing Science Department University of Glasgow (Nov. 86)

[LANDA 83] LANDAUER, T.K., GALOTTI, K.M. & HARTWELL, S. *"Natural Command Names and Initial Learning: A Study of Text-Editing Terms"* CACM Vol.**26** Num.7 pp495-503 (1983)

[LANGR 74] LANGRIDGE, R. *"Interactive Three-Dimensional Computer Graphics in Molecular Biology"* Federation Proceedings Vol.**33** Num.12 (1974)

[LANGR 81] LANGRIDGE, R., FERRIN, T.E., KUNTZ, I.D. & CONNOLLY, M.L. *"Real-Time Color Graphics in Studies of Molecular Interactions"* Science Vol.**211** pp. 661-666 (1981)

[LANGR 84] LANGRIDGE, R., FERRIN, T.E., KUNTZ, I.D. & CONNOLLY, M.L. *"The Future of Molecular Graphics."* Journal of Molecular Graphics Vol.**2** Num.2, pg. 56 (1984)

[LEE 71] LEE, B. & RICHARDS, F.M. *"The Interpretation of Protein Structures: Estimation of Static Accessibility"* Journal of Molecular Biology Vol.**55** pp. 379-400 (1971)

[LESK 82] LESK, A.M. & HARDMAN, K.D. *"Computer-Generated Schematic Diagrams of Protein Structures"* Science, Vol.**216** pp.539-540 (1982)

[LESK 84] LESK, A.M. *"Themes and contrasts in protein structures"* Trends Biochem.Soc. pg V (1984)

[LESK 85] LESK, A.M. & HARDMAN, K.D. *"Computer-Generated Pictures of Proteins"* Methods in Enzymology Vol.**115** pp.381-390 (1985)

[LEVIN 66] LEVINTHAL, C. *"Molecular Model-building by Computer."* Scientific American. Vol.**214** Num.6 pp.42-52 (Jun.1966)

[LEVIT 77] LEVITT, M. & GREER, J. *"Automatic Identification of Secondary Structure in Globular Proteins"* Journal of Molecular Biology Vol.**114** pp. 181-239 (1977)

[LEWIS 71] LEWIS, P.N., MOMANY, F.A. & SCHERAGA,H.A. *"Folding of Polypeptide Chains in Proteins: A Proposed Mechanism for Folding"* Proc. Natl. Acad. Sci. U.S.A. Vol. **68** pp. 2293-2297 (1971)

[LIFSO 80] LIFSON, S. & SANDER, C. *"Specific Recognition in the Tertiary Structure of β–Sheets of Proteins"* Journal of Molecular Biology Vol.**139** pp. 627-639 (1980)

[MEYER 74] MEYERS, E.F. *"Storage and Retrieval of Macromolecular Structural Data"* Biopolymers Vol.**13**, pp.419-424 (1974)

[MILLE 79] MILLER, J.R. *"A computer Graphics System for Macromolecular Model Building"* Ph.D. Thesis. Purdue University (1979)

[MILNE 85] MILNER-WHITE, E.J. & POET, R.R. *"Computer Graphics of Larges Macromolecules."* Biochemical Society Transactions. Vol.**13** pp.793-795 (1985)

[MILNE 87a] MILNER-WHITE, E.J. *"Beta-Bulges within Loops as Recurring Features of Protein Structure"* Biochemica Biophysica Acta Vol. **911** pp. 261-265 (1987)

[MILNE 87b] MILNER-WHITE, E.J. & POET, R.R. *"Loops, Bulges, Turns and Hairpins in Proteins"* Trends in Biochemical Science Vol.**12** Num.6 pp. 189-192 (1987)

[MILNE 88a] MILNER-WHITE, E.J. *"Recurring Loop Motif in Proteins that Occurs in Rigth-Handed and Left-Handed Forms"* Journal of Molecular Biology Vol.**199** pp.503-511 (1988)

[MILNE 88b] MILNER-WHITE, E.J., ROSS, B.M, ISMAIL, R., BELHADJ-MOSTEFA, K. & POET, R.R. *"One Type of Gamma-Turn Rather Than the Other Gives Rise to Chain-Reversal in Proteins"* Journal of Molecular Biology Vol.**204** pp.777–782 (1988)

[MILNE 90] MILNER-WHITE, E.J., BELHADJ-MOSTEFA, K., POET, R.R. & WRIGHT, C.S. *"Hydrogen Bond Conservation in Proteins"* Not yet Published (1990)

[MORFF 83a] MORFFEW, A.J. *" Where Do We Go from Here? A Personal View on the Future of Molecular Graphics"* Journal of Molecular Graphics Vol.**1** Num.3 pp. 83-87 (1983)

[MORFF 83b] MORFFEW, A.J., TODD, S.J.P. & SNELGROVE, M.J. *"The Use of a Relational Data Base for Holding Molecular Data in a Molecular Graphics System"* Computer & Chemistry Vol.**7** pp. 9-16 (1983)

[MORFF 84] MORFFEW, A.J. *"Editorial: Databases in Molecular Graphics"* Journal of Molecular Graphics Vol.**2** Num.3 pp. 66-69 (1984)

[MORFF 86] MORFFEW, A.J. & TODD, S.J.P. *"The Use of PROLOG as a Querying Language"* Computer & Chemistry Vol.**10** pp. 9-14 (1986)

[MORRI 88] MORRIS, G.M. *"The Matching of Protein Sequence using Color Intrasequence Homology Displays"* Journal of Molecular Graphics Vol.6 Num.3 pp. 135-140 (1988)

[MOTHE 78] MOTHERWELL, S. *"Program for Plotting Molecular Structure Illustrations"* -PLUTO78- Crystallographic Data Centre, University Chemical Laboratory, Cambridge, UK (Avril78)

[NEEDL 70] NEEDLEMAN, S.B. & WUNSCH, C.D. *"A General Method Applicable to the Search for Similarities in the Amino Acid Sequence of Two Proteins"* Journal of Molecular Biology, Vol.**48** pp. 443-453 (1970)

[NEWEL 72] NEWELL, M.E., R.G. NEWELL & SANCHA, T.L. *" A New Approach to the Shaded Picture Problem"* Proc. ACM Nat. Conf. pg.443 (1972)

[NEWMA 79] NEWMAN, W.M & SPROULL, R.F *"Principles of Interactive Computer Graphics"* McGraw-Hill Book Company (1979)

[OPHIR 69] OPHIR, D., SHEPHERD, B.J. & SPRINRAD, R.J. *"Three-Dimensional Computer Display"* Communications of the ACM, Vol.**12,** Num.6 (1969)

[PAULI 51] PAULING, L., COREY, R.B. & BRANSON, H.R. *"The Structure of Proteins: Two Hydrogen-Bonded Helical Conformations of the Polypeptide Chain"* Proc. Natl Acad. Sci. U.S.A. Vol.**37** pp.205-211 (1951)

[PERUT 64] PERUTZ, M.F. *"The Haemoglobin Molecule".* Scientific American pp. 64-76 (1964)

[POET 86] POET, R.R. & MILNER-WHITE, E.J.*"Displaying Relevant Features of Protein Molecules."* Computer Graphics Forum Vol.**5** Num.3 pp. 211-216 (1986)

[QUARE 84] QUARENDON, P. *"The Use of Constructive Solid Geometry for Molecular Graphics"* Journal of Molecular Graphics Vol.**2** Num.2 pp. 59-60 (1984)

[RICHA 77] RICHARDS, F.M. *"Areas, Volumes, Packing, and Protein Structure"* Annu. Rev. Biophys. Bioeng. Vol.**6** pp. 151-176 (1977)

[RICHA 81] RICHARDSON, J.S. *"The Anatomy and Taxonomy of Protein Structure"* Advances in Protein Chemistry Vol.**34** pp. 167-339 (1981)

[ROSE 76] ROSE, G.D., WINTERS, R.H. & WETLAUFER, D.B. *"A Testable Model for Protein Folding"* FEBS Letters Vol **63** pp 10-16 (1976)

[SALMO 87] SALMON, R. & SLATER, M *"Compter Graphics: Systems & Concepts"* Addison-Wesley Publishing Company, Inc, (1987)

[SCHEI 86] SCHEIFER, R.W. & GETTYS, J. *"The X Window System"* A C M Transactions on Graphics Vol.**5** Num.2 pp. 79-109(1986)

[SCHEI 88] SCHEIFER, R.W., GETTYS, J. & NEWMAN, *"X Window System - C Library and Protocol Reference "* Digital Press (1988)

[SHNEI 87] SHNEIDERMAN, B. *"Designing the User Interface."* Addison-Wesley Publishing Company (1987)

[STERN 87] STERN, H.L. *"Comparison of Window Systems"* BYTE Nov.87 pp. 265–272 (1987)

[STERNB 83] STERNBERG, M.J.E. *"The Analysis and Prediction of Protein Structure"* Computing in Biological Science. Elseviour Biomedical Press pp.143-177 (1987)

[STRYE 75] STRYER, L. *"Biochemistry"* W.H. Freeman and Company San Franscisco pp.11-69 (1975)

[TAYLO 87] TAYLOR, W.R. *"Protein Structure Prediction"* Synthesis and Applications of DNA and RNA, Chapt. 12, Orlando (1987)

[THORN 81] THORNTON, J.M. *"Disulphide Bridges in Globular Proteins"* Journal of Molecular Biology Vol.**151** pp.261-287 (1981)

[THORN 88] THORNTON, J.M., SINGH, J., CAMPBELL, S.F. & BLUNDELL, T.L. *"Protein-Protein Recognition via Side-Chain Interaction"* Biochem. Soc. Trans. Vol.**16** pp.927-930 (1988)

[TODD 83] TODD, S.J.P. & GILLET, J. *"Animation in the Winchester Graphics System"* Journal of Molecular Graphics Vol.**1** Num.2 pp. 39-42 (1983)

[TOMA 87a] TOMA, K. *"Simple Protein Model Building Tool"* Journal of Molecular Graphics Vol.**5** Num.2 pp. 101-102 (1987)

[TOMA 87b] TOMA, K. *"Investigation of Protein Inter- and Intramolecular Interactions with a Simple Graphics Tools"* Journal of Molecular Graphics Vol.5 Num.3 pp. 149-151 (1987)

[WHITE 84] MG-1 Owner Operator Guide *"Owner Operator Manual"* (1984)

[WRIGH 87] WRIGHT, C.S. *"Refinement of the Crystal Structure of Wheat Germ Agglutinin Isolectin 2 at 1.8 Angstroms Resolution"* Journal of Molecular Biology. Vol.**194** pp. 501-529 (1987)